VO „Multimedia Security“

A. Uhl
Multimedia Signal Processing and Security Lab (WaveLab)
Fachbereich Computerwissenschaften
Universität Salzburg

17. Januar 2011

Personalia: A. Uhl

Email-Adresse: uhl@cosy.sbg.ac.at.

Basis-URL: http://www.cosy.sbg.ac.at/~uhl.

Büro: FB Computerwissenschaften, Zi. 1.11, Jakob-Haringer Str. 2, Salzburg-Itzling.

Telefonnummer (Büro): (0662) 8044-6303.

Telefonnummer (Sekretariat): (0662) 8044-6328 oder -6343.

Formalia

LVA-URL: http://www.cosy.sbg.ac.at/~uhl/student.html.

Abhaltezeit der LVA: Di 11:30 - 13:00

Termine: wöchentlich

Abhalteort der LVA: Hörsaal T05

Vorwort

Willkommen zur Lehrveranstaltung MULTIMEDIA SECURITY. Es handelt sich um eine Überblicksveranstaltung die sich aber trotzdem nahe an aktuellen Forschungsthemen bewegt, da das Gebiet als solches sehr jung ist.

Multimedia Security hat sich zu einem recht großen Gebiet entwickelt, die Ausrichtung dieser LVA richtet sich weitgehend nach meinen eigenen Forschungskompetenzen, die im Bereich Multimedia Signalverarbeitung angesiedelt sind (Schwerpunkt Verarbeitung von visuellen Daten).

Top Zeitschriften:

- IEEE Transactions on Information Forensics and Security (TIFS)
- EURASIP Journal on Information Security
- Springer LNCS Transactions on Data Hiding and Multimedia Security

.... oder in allgemeinen Zeitschriften im Bereich Multimedia, Signalverarbeitung und Security (“CRYPTOGRAPHY MARRIES MULTIMEDIA SIGNAL PROCESSING”).

Hintergrund - Tagungen

Wichtigste Tagungen zu diesem Themenkreis:

- ACM Multimedia and Security Workshop (2010 in Rom, 2009 in Princeton, 2008 in Oxford mit jeweils 2 Arbeiten)
- SPIE's Media Forensics and Security (immer im Rahmen von Electronic Imaging in San Jose)
- Int. Workshop on Digital Watermarking (2010 in Seoul, 2009 in Surrey, 2008 in Busan mit je 1 Arbeit)
- Information Hiding (2010 in Calgary, 2009 in Darmstadt, 2008 in Santa Barbara)
- Communications and Multimedia Security CMS (2010 in Linz mit 4 Arbeiten, 2011 in Ghent mit AU als PC)
- IEEE International Workshop on Information Forensics and Security WIFS (2009 in London, 2010 in Seattle)

..... und auf grösseren Tagungen im Bereich Multimedia, Signalverarbeitung, Information Security.

Hintergrund - Projekte

Die folgenden Projekte wurden/werden in den letzten Jahren am FB Computerwissenschaften im Bereich Multimedia (Sicherheit) abgewickelt:

- Adaptive Security Techniques for Visual Data in Wavelet-based Representation (FWF, 100K EUR)
- Object-based Image and Video Compression with Wavelet Techniques (FWF, 130K EUR)
- Wavelet Compression for Video Surveillance Applications (Dallmeier Elektronik, 120K EUR)
- Adaptive Streaming of Secure Scalable Wavelet-based Video (FWF, 230K EUR)
- Sample Data Compression and Encryption in Biometric Systems (FWF, 210K EUR)
- ECRYPT - IST European Network of Excellence in Cryptology Watermarking Virtual Lab (EU, 50K EUR)

Hintergrund - Literatur

- Monographs
  - ⋆ B. Furht and D. Kirovski. Multimedia Security Handbook. CRC Press, 2005.
  - ⋆ W. Zeng, H. Yu, and C.-Y. Lin. Multimedia Security Techniques for Digital Rights Management. Elsevier Science, 2006.
  - ⋆ A. Uhl and A. Pommer. Image and Video Encryption: From Digital Rights Management to Secure Personal Communications. Springer Series on Advances in Information Security vol. 15, 2005.
  - ⋆ S. Katzenbeisser and F. Peticolas. Information Hiding Techniques for Steganography and Digital Watermarking. Artec House, 1999.
  - ⋆ I. Cox, M. Miller, and J. Bloom. Digital Watermarking and Steganography. Academic Press, 2008.
  - ⋆ N. Johnson, Z. Duric, and S. Jajodia. Information Hiding: Steganography and Watermarking - Attacks and Countermeasures, Kluwer Series on Advances in Information Security, 2000.
  - ⋆ R. Liu, W. Trappe, J. Wang, M. Wu, and H. Zhao. Multimedia Fingerprinting Forensics for Traitor Tracing. EURASIP Series on SP&V, Vol. 4, 2005.
  - ⋆ More monographs on Watermarking available !

Hintergrund - Lokale Beiträge

- Dominik Engel, Thomas Stütz, Andreas Uhl. A survey on JPEG2000 encryption. ACM/Springer Multimedia Systems 15:4, 243-270, 2009.
- Dominik Engel, Thomas Stütz, Andreas Uhl. Format-compliant JPEG2000 Encryption in JPSEC: Security, Applicability and the Impact of Compression Parameters. EURASIP Journal on Information Security, Article ID 94565, pp. doi:10.1155/2007/94565, 20 pages, 2007.
- Hermann Hellwagner, Robert Kuschnig, Thomas Stütz, Andreas Uhl. Efficient In-Network Adaptation of Encrypted H.264/SVC Content. Signal Processing: Image Communication 24:9, 740-758, 2009.
- Peter Meerwald, Christian Koidl, Andreas Uhl. Attack on 'Watermarking Method Based on Significant Difference of Wavelet Coefficient Quantization'. IEEE Transactions on Multimedia 11:5, 1037-1041, 2009.
- Gerold Laimer, Andreas Uhl, Key Dependent JPEG2000-Based Robust Hashing for Secure Image Authentication, EURASIP Journal on Information Security, Article ID 895174, doi:10.1155/2008/895174, 19 pages, 2008
- Dominik Engel, Elias Pschernig, Andreas Uhl. An Analysis of Lightweight Encryption Schemes for Fingerprint Images. IEEE Transactions on Information Forensics and Security 3:2, pp. 173-182, 2008.

Inhalte

Die VO wird sich insbesondere mit folgenden Problemkreisen beschaeftigen:

? Kryptographie Primer ?

Verschlüsselung von Visuellen Daten

Robust Watermarking: DRM (Digital Rights Management), Annotation, Copy Control, Fingerprinting, ...

Steganographie

(Semi)Fragiles Watermarking

Visual / Perceptual / Robust Hashing

Media Forensics

? Biometrie ?

Sonstiges

- Prüfung: Lausur ODER Anwesenheit zu den Vorlesungsterminen mit Zusatzleistungen (Projekttechnisch oder spezielle Papers)

- Skriptum (Slides) auf meiner Webseite `www.cosy.sbg.ac.at/~uhl/student.html` zum Download

Inhalt

- Kryptographie Primer,
- Verschlüsselung von Visuellen Daten,
- Information Hiding & Watermarking: DRM,
- Integritätsprüfung für Multimediale Daten.
- Forensische Methoden für Multimediale Daten.

Table of Contents: Kryptographie Primer

- Terminologie
- Symmetrische Algorithmen,
- Public-Key Algorithmen,
- Attacken gegen Krypto-Systeme,
- Protokolle,
- Digitale Unterschriften,
- DES, AES, RSA, MD-5, DSA.
- IPSec
- DNSSec

Kryptographie Primer

- Terminologie
- Symmetrische Algorithmen,
- Public-Key Algorithmen,
- Attacken gegen Krypto-Systeme,
- Protokolle,
- Digitale Unterschriften,
- DES, AES, RSA, MD-5, DSA.
- IPSec
- DNSSec

Terminologie I

- Sender - Empfänger: Ein Sender möchte eine Nachricht so senden, daß ein potentieller Lauscher diese nicht lesen kann. Eine Nachricht ist ein Text, deren Inhalt zugänglich ist und heißt auch Plain Text oder Clear Text. Plain Text wird verschlüsselt (encryption), die verschlüsselte Nachricht heißt dann Ciphertext. Ciphertext muß vor dem Lesen entschlüsselt werden (decription) (encipher - decipher).

- Kryptographie - Kryptologie: Kryptographie ist die Kunst bzw. Wissenschaft Nachrichten sicher zu verschlüsseln. Kryptologie ist ein mathematisches Teilgebiet, das sich mit Kryptographie und Kryptoanalyse beschäftigt.

- Plaintext - Ciphertext - Encryption: E(M)=C mit E Encryption, C Ciphertext und M Message/Plaintext.

$$\Rightarrow D(E(M)) = M!!$$

Terminologie II

- Confidentiality: klassische Verschlüsselung
- Authentication (Authentifizierung): eindeutige Identifizierung einer Person oder eines Gerätes
- Integrity (Integrität): Nachricht wurde beweisbar nicht verändert
- Nonrepudiation: eine Handlung/Tatsache kann nicht fälschlicherweise geleugnet werden.
- Kerckhoff Prinzip: die gesamte Sicherheit eines Kryptografie-Verfahrens soll nur auf der Geheimhaltung des Schlüssels beruhen und nicht auf der Geheimhaltung des kryptografischen Algorithmus (Auguste Kerckhoff, 1835-1903).

Symmetrische Algorithmen

Bei diesen Algorithmen kann der Verschlüsselungskey aus dem Entschlüsselungskey berechnet werden und umgekehrt. Meist sind sie sogar identisch.

Bei den sogenannten “One Key Algorithms müssen sich SenderIn und EmpfängerIn auf einen gemeinsamen Key einigen. Dazu müssen sie sicher kommunizieren können. Die Sicherheit dieses Systems liegt in der Sicherheit des Schlüssels, der geheim bleiben muß.

Grundsätzlich können zwei Kategorien unterschieden werden:

- Stream Ciphers: arbeiten zu einem bestimmten Zeitpunkt t an einem einzelnen Bit (manchmal auch Byte).
- Block Ciphers: arbeiten zu einem bestimmten Zeitpunkt t an einer Gruppe von Bits. Typischerweise sind dies 64 - 128 Bits.

Probleme bei symmetrischen Algorithmen

- Die Schlüsselverteilung muß geheim sein. Das ist bei großen Systemen natürlich problematisch.

- Wenn der Key verloren geht, kann etwa Eve Bob vortäuschen, Alice zu sein und falsche Nachrichten verschicken.

- Wenn in einem Netzwerk für jedes AnwenderInnen Paar ein verschiedener Key verwendet, ergeben sich $\frac{n(n-1)}{2}$ Keys, mit $n \ldots$#User. Das ist natürlich nur bei einer kleinen Zahl von AnwenderInnen praktikabel.

Public-Key Algorithmen

Diese Klasse von Algorithmen werden auch als asymmetrische Verfahren bezeichnet. Der Verschlüsselungskey und der Entschlüsselungskey sind nicht identisch. Der Entschlüsselungskey kann nicht (in vernünftiger Zeit) aus dem Verschlüsselungskey berechnet werden.

- Public Key: Der Verschlüsselungskey kann öffentlich bekannt gegeben werden und wird daher auch als Public Key bezeichnet. Jede Person, die Zugang zu diesem Schlüssel hat, kann eine Nachricht verschlüsseln. Eine Entschlüsselung ist damit aber nicht mehr möglich.

- Priviate Key: Der Entschlüsselungskey muß geheim gehalten werden und wird daher auch als Private Key bezeichnet. Jene Person, die ihn besitzt, kann eine mit dem zugehörigen Public Key verschlüsselte Nachricht entschlüsseln.

Diese Verfahren werden auch für digitale Unterschrifen verwendet. In diesem Fall erfolgt die Verschlüsselung mit dem Private Key, die Entschlüsselung mit dem Public Key.

Public-Key Algorithmen haben den großen Vorteil, daß kein sicherer Kanal für die bertragung eines gemeinsamen Schlüssels benötigt wird.

One-Way Functions

One-Way Functions sind ein wesentlicher Bestandteil der Public-Key Kryptographie und verwenden Funktionen mit folgenden Eigenschaften:

- Bei gegebenem x ist es leicht $f(x)$ zu berechnen.
- Bei gegebenem $f(x)$ ist es sehr aufwendig x zu berechnen (=entschlüsseln).

Dieses Verfahren hat aber folgendes **Problem**: Nicht nur eine Attacke sondern auch das autorisierte Entschlüsseln wird damit erschwert bzw. unmöglich gemacht.

Daher wurden einige etwas modifizierte Verfahren entwickelt: Trapdoor One-Way Functions. Das sind Funktionen mit den bereits beschriebenen Eigenschaften und zusätzlich mit einem "Geheimnis" Y. Wer immer Y kennt, kann x schnell aus $f(x)$ berechnen. One-Way Hash Functions sind sog. Kompressions- und Kontraktions Funktionen, Fingerprints usw. Als Input wird ein pre-image mit variabler Länge benutzt. Die Funktion berechnet einen Output mit fixer Länge (hash value). Ein Message Authentication Code (MAC) ist eine One-Way Hash Funktion mit einem geheimen Key.

Kommunikation mit Public-Key Kryptographie

Jeder, der den Public Key besitzt, kann eine Nachricht verschlüsseln aber nicht mehr entschlüsseln. Nur jene Person, die den Private Key besitzt, kann die Nachricht wieder entschlüsseln. Die mathematische Grundlage hierfür sind Trapdoor One-Way Functions. Verschlüsseln bedeutet $f(x)$ aus x zu berechnen. Entschlüsseln ist die Berechnung von x aus $f(x)$, das "Geheimnis" Y ist der Private Key.

Public-key Verfahren haben ein viel einfacheres Schlüsselmanagement. In der Praxis wird das meist so gehandhabt, daß zuerst ein System festgelegt wird. Jede/r TeilnehmerIn besitzt einen Private und einen Public Key. Der Public Key liegt in einer für alle zugänglichen Datenbank. Alice kann sich dann den Public Key von Bob in dieser Datenbank selbst abholen. Dabei muß garantiert sein, daß ein in der Datenbank deponierter Public Key auch wirklich von der richtigen Person stammt. Hierfür gibt es je nach Anzahl und geographischer Verteilung der TeilnehmerInnen verschiedene Methoden, z.B. Web of Trust, Notariate,....

Hybride Systeme

Public Key Systeme sind KEIN Ersatz für symmetrische Algorithmen:

- Public Key Systeme sind langsamer (etwa um den Faktor 1000).
- Sie sind durch Chosen-Plaintext Attacken verwundbar

Daher wird in den meisten praktischen Einsatzbereichen wird Public Key Kryptographie verwendet, um Keys eines symmetrischen Verfahrens verschlüsselt zu verteilen:

- Bob schickt Alice seinen Public Key.
- Alice erzeugt einen zufälligen Session Key K, verschlüsselt diesen mit Bobs Public Key und schickt in zu Bob. $E_B(K)$.
- Bob entschlüsselt Alices Nachricht mit seinem Private Key und erhält den Session Key. $D_B(E_B(K)) = K$.
- Beide ver- und entschlüsseln ihre Kommunikation mit dem Session Key.

Attacken gegen kryptographische Systeme I

Ciphertext-only Attack: Der/die FeindIn besitzt den Ciphertext von verschiedenen Nachrichten, die mit dem gleichen Algorithmus verschlüsselt wurden.
geg: $C_1, \ldots C_i$
ges: $P_1, \ldots P_i$ oder Algorithmus um P_{i+1} aus C_{i+1}berechnen zu können.

Known Plaintext Attack: Der/die FeindIn kennt Ciphertexte und Plaintexte.
geg: $P_1, C_1, \ldots P_i, C_i$
ges: Key oder Algorithmus um P_{i+1} aus C_{i+1}berechnen zu können.

Chosen Plaintext Attack: Der/die FeindIn kann den Plaintext auswählen, der verschlüsselt wird. Dadurch können bestimmte Plaintexts verwendet werden, die mehr Information über die Schlüssel liefern.
geg: $P_1, C_1, \ldots P_i, C_i$, wobei $P_1, \ldots P_i$ frei wählbar sind
ges: Key oder Algorithmus um P_{i+1} aus C_{i+1} berechnen zu können.

Attacken gegen kryptographische Systeme II

Adaptive-Chosen Plaintext Attack: Der/die FeindIn kann den Plaintext adaptiv verändern, je nach Ergebnis des letzten Ver- und Entschlüsselungsprozesses.

Chosen-Ciphertext Attack: Der/die FeindIn kann verschiedene Ciphertexte auswählen, die entschlüsselt werden und hat Zugang zum entschlüsselten Plaintext. Diese Attacke wird vor allem bei Public Key Systemen verwendet. geg: C_1, P_1, $\ldots C_i$, P_i, wobei $P_1, \ldots P_i$ frei wählbar sind
ges: Key

Rubber-Hose Cryptoanalysis: Verwendung von Folter und Bedrohung um einen Key zu bekommen. Das ist oft am effizientesten.

One-Time Pads

Große, sich nicht wiederholende Menge von wirklich zufälligen Key-Buchstaben.

Die Verschlüsselung erfolgt dann auf folgende Weise:
$\oplus 26$ des Plaintext und One-Time Pad. Jedes Keyelement wird nur einmal in einer Nachricht verwendet. Bei einer neuen Nachricht muß ein neuer Key benutzt werden. Das soll die perfekte Sicherheit gewährleisten. Ein zufälliger Key macht aus einem nicht-zufälligen Plaintext einen zufälligen Ciphertext.

- Attacke:
  - ⋆ Eine mögliche Attacke richtet sich gegen das zufällige Erzeugen von Keys (PRNG)!
  - ⋆ Problem: Key darf nie wieder verwendet werden. Für bits: analog mit XOR
- Probleme: Diese Methode ist nur für kleine Nachrichten geeignet. Das Senden und Rekonstruieren muß synchronisiert werden.
- Anwendung: “ultra secure low-bandwidth (z.B. “Rotes Telefon USA - Moskau)

Protokolle

Mitspieler: Alice, Bob, Carol, Dave, Eve, Mallory, Trent

- Arbitrated Protocol: Ein Vertrauensmann wickelt so ein Protokoll ab. Im Netz ist es schwierig jemanden vertrauenswürdigen zu finden, ist ein Kommunkationsbottleneck, Frage nach Finanzierung !

- Adjudicated Protocol: Zweite Ebene, wird nur invoked wenn ein Problem auftritt, nach dem Motto “nicht Betrug verhindern sondern aufdecken”.

- Self Enforcing Protocol: Das Protokoll selbst garantiert die Fairness.

Attacken gegen Protokolle

- Passive Attack: Eve kann das Protokoll mithören. Entspricht einer Ciphertext-only attack.
- Active Attack: Mallory verändert das Protokoll zu seinem Vorteil (Er täuscht vor, ein anderer zu sein, erfindet neue Nachrichten im Protokoll, löscht, ersetzt, usw . . .)
- Passive Cheating: Die AkteurInnen dieser Attackie sind diverse ProtokollteilnehmerInnen, die zwar dem Protokoll folgen, aber mehr Information wollen, als ihnen zusteht.
- Active Cheating: Die AkteurInnen dieser Attackie sind diverse ProtokollteilnehmerInnen, die während der Teilnahme das Protokoll ändern.

Digitale Unterschriften I

Warum sind Unterschriften so wichtig?

- Eine Unterschrift ist authentisch und identifizeirt die/den Unterzeichnende/n (Unterschreiber ist wer er vorgibt zu sein),
- nicht fälschbar (nicht von jemand anderem imitierbar),
- nicht wiederverwendbar, z.B. auf einem anderen Dokument.
- Das unterschriebene Dokument kann nicht verändert werden.
- Die/der Unterschreibende kann die Unterschrift nicht leugnen.

Diese Aufzählung entspricht natürlich etwas mehr dem Wunschdenken als der Wirklichkeit. Am Computer existieren weitere Probleme: Dateien können kopiert werden, Unterschriften können mit Cut und Paste auf ein neues Dokument “geklebt werden, Dateien können leicht modifiziert werden, ohne daß es jemand bemerkt.

Digitale Unterschriften II: Grundidee

- Alice erzeugt den One-Way Hash eines Dokumentes.
- Alice verschlüsselt den Hash mit ihrem Private Key (=Unterschrift).
- Aice schickt das Dokument mit dem Unterschriftshash an Bob.
- Bob produziert One-Way Hash des Dokumentes und entschlüsselt den unterschriebenen One-Way Hash. Stimmen die beiden überein, ist die Unterschrift gültig.

Digitale Unterschriften III: mit Verschlüsselung

- Alice unterschreibt m mit ihrem Private Key $S_A(m)$.
- Alice verschlüsselt unerschriebenes m mit Bobs Public Key und schickt ihm $E_B(S_A(m))$.
- Bob entschlüsselt m mit seinem Private Key $D_B(E_B(S_A(m))) = S_A(m)$.
- Bob überprüft die Unterschrift von Alice mit ihrem Public Key $V_A(S_A(m)) = m$.

Somit ergibt sich also eine Unterschrift sowohl auf dem Umschlag als auch auf dem Brief. Es werden für das Verschlüsseln und Unterschreiben verschieden Schlüsselpaare verwendet (außerdem: timestamps).

Data Encryption Standard DES

Anfang der 70er Jahre von IBM entwickelt (Lucifer), 1976 standardisiert, 1987, 1993 und 1998 wiederzertifiziert. Jedoch sicherheitstechnisch nicht mehr auf dem Stand der Zeit.

DES ist ein symmetrisches Blockcipher Verfahren mit 64 Bit großem Block und 64 (eigentlich 56) Bit Schlüssel. Nach einer initialen Permutation wird der Block in zwei Hälften geteilt (je 32 Bit). In den 16 Runden identischer Operationen werden die Daten mit dem Schlüssel kombiniert. Am Ende werden die beiden Hälften wieder zusammengeführt und eine finale Permutation beendet den Algorithmus.

In jeder Runde wird der Schlüssel geshiftet und von 56 Bit 48 ausgewählt. Die rechte Hälfte der Daten wird auf 48 Bits vergrößert (Expansion, Permutation) und ein XOR auf diese Daten und den geshifteten und permutierten Schlüssel angewendet. Die resultierenden Daten werden durch 8 sogenannte S-Boxen geschickt (= 32 Bits) und dann nochmals permutiert. Dieser gesamte Vorgang wird mit der Funktion f dargestellt. Auf das Ergebnis von f und die linke Hälfte der Daten wird dann ein XOR angewendet. Das Ergebnis dieser Operation wird dann die neue rechte Hälfte. Die alte rechte Hälfte wird die neue linke Hälfte.

Der gesamte bis jetzt beschriebene Vorgang wird 16 mal wiederholt.

Betriebsmodi von Blockciphern I

- Electronic Codebook Mode (ECM): Ein Block Plaintext wird in einen Block Ciphertext überführt. Dadurch ergeben sich diverse Vorteile aber auch Probleme.
  - ⋆ Vorteile: Reihenfolge der Blöcke muß bei der Bearbeitung nicht eingehalten werden (Datenbanken, parallele Verarbeitung). Keine Fehlerforpflanzung über Blockgrenzen.
  - ⋆ Nachteile: Attacken durch Bildung von Plain-Ciphertextpaaren (Textbeginn, -ende), Block replay

- Cipher Block Chaining Mode (CBC): Ein zusätzlicher Feedback Mechansimus wird eingefügt. Die Ergebnisse der Verschlüsselung des vorherigen Blocks werden zur Verschlüsselung des aktuellen Blocks verwendet. Jeder Ciphertext Block hängt nicht nur vom entsprechenden Plaintext Block ab, sondern auch von allen vorherigen Plaintext Blöcken.

Betriebsmodi von Blockciphern II

- CBC (cont.): Auf den Plaintext Block wird mit dem vorhergehenden Ciphertext Block ein XOR angewendet, ehe er verschlüsselt wird.

$$C_i = E_K(P_i \oplus C_{i-1})$$

$$P_i = C_{i-1} \oplus D_K(C_i)$$

  - ⋆ Probleme beim CBC: Zwei identische Texte werden gleich verschlüsselt. Wenn sie sich ab einem bestimmten Punkt unterscheiden, unterscheiden sich auch die beiden Ciphertexte an der genau gleichen Stelle. Abhilfe schafft ein zufälliger Initialisierungsvektor als erster Block

Betriebsmodi von Blockciphern III

- Cipher Feedback Mode (CFB): Block Ciphers werden auf diese Weise auch als "self-synchronizing stream ciphers verwendet. Im CBC Mode muß gewartet werden, bis der gesamte Block vorhanden ist. Bei dieser Variante genügt es, wenn n Bits zur Verfügung stehen.

  Ein Block Algorithmus im CFB Mode arbeitet mit einer Queue die genau Blockgröße hat. Zu Beginn wird die Queue mit einem Initialisierungsvektor gefüllt. Die Queue wird verschlüsselt und auf die n-Bit am linken Rand der Queue wird mit dem Plaintext ein XOR angewendet und an das rechte Ende der Queue gestellt. Alle anderen Bits der Queue werden nach links verschoben. Dann kommt der nächste n-Bit Block an die Reihe.

- Output Feedback Mode (OFB): hier werden n-Bits des verschlüsselten Shift Registers wieder im Shift Register verwendet. Die Entschlüsselung erfolgt wieder genau umgekehrt zur Verschlüsselung. Hier ist der Feedback u. a. von Plain- und Ciphertext, wird auch als "internal feedback" bezeichnet.

  Der gesamte Verschlüsselungsvorgang kann geschehen, bevor der Plaintext vorhanden ist, da nur noch ein XOR angewendet wird.

Sicherheit von DES

- Parameter: Blockgröße zu klein, ebenso die Schlüssellänge

- Anzahl der Runden: Jeder DES mit weniger als 16 Runden kann mit “Differential Cryptoanalysis weitaus effizienter als mit einer Brute Force Attacke gebrochen werden.

- Algebraische Struktur: 64-Bit Plaintext Blocks ergeben 64 Bit Ciphertext Blöcke auf 2^{64} verschiedene Arten. DES benutzt 2^{56} (etwa 10^{17}) davon.

  Mit mehrfacher Verschlüsselung scheint es möglich zu sein, einen größeren Anteil auszunutzen.

  Ist DES eine abgeschlossene Gruppe (im algebraischen Sinn), gibt es für jedes Schlüsselpaar K_1 und K_2 ein K_3, sodaß

$$E_{K_2}(E_{K_1}(P)) = E_{K_3}(P)$$

  1992 wurde allerdings bewiesen, daß DES keine Gruppe darstellt. Das heißt, Mehrfachverschlüsselung macht Sinn.

DES Varianten

- Double Encription: durch “Meet in the middle Attacke nicht sicherer als einfache Verschlüsselung.

- Triple Encription: nur mit drei Keys wirklich verbesserte Sicherheit. EDE Modus unsicher. Outer CBC ist die sicherste Variante:

$$C_i = E_{K_3}(D_{K_2}(E_{K_1}(P_i \oplus C_{i-1})))$$

$$P_i = C_{i-1} \oplus D_{K_1}(E_{K_2}(D_{K_3}))$$

Advanced Encryption Standard AES

AES ist der offizielle Nachfolger von DES. In einem offenen Ausschreibungsverfahren wurde aus 15 Submissions durch ein Expertenkommittee der “beste” Algorithmus ausgewählt. Juni 1998 AES Submission deadline, August 1999 wurden die 5 Finalisten bekanntgegeben, im Oktober 2000 stand der an der KU Leuven (Belgien) entwickelte “Rijndael” cipher als AES fest. Die Entscheidung ist ein Kompromiss zwischen Sicherheit und Ausführungseffizienz. Der Blockcipher “Square” ist der Vorgänger dessen bekannte Schwächen ausgebessert wurden.

AES ist ein klassischer Blockcipher mit 128 Bits Blockgrösse (Rijndael erlaubt auch 192 und 256 Bits), mit Schlüssellänge 128, 192 oder 256 Bits und einer variablen Anzahl von Runden (9 bei 128 Bits Key, 11 bei 192 Bits Key und 13 bei 256 Bits Key). AES operiert auf Byteebene. Eine einzelne Runde von AES besteht aus folgenden Elementen.

AES Funktionsweise

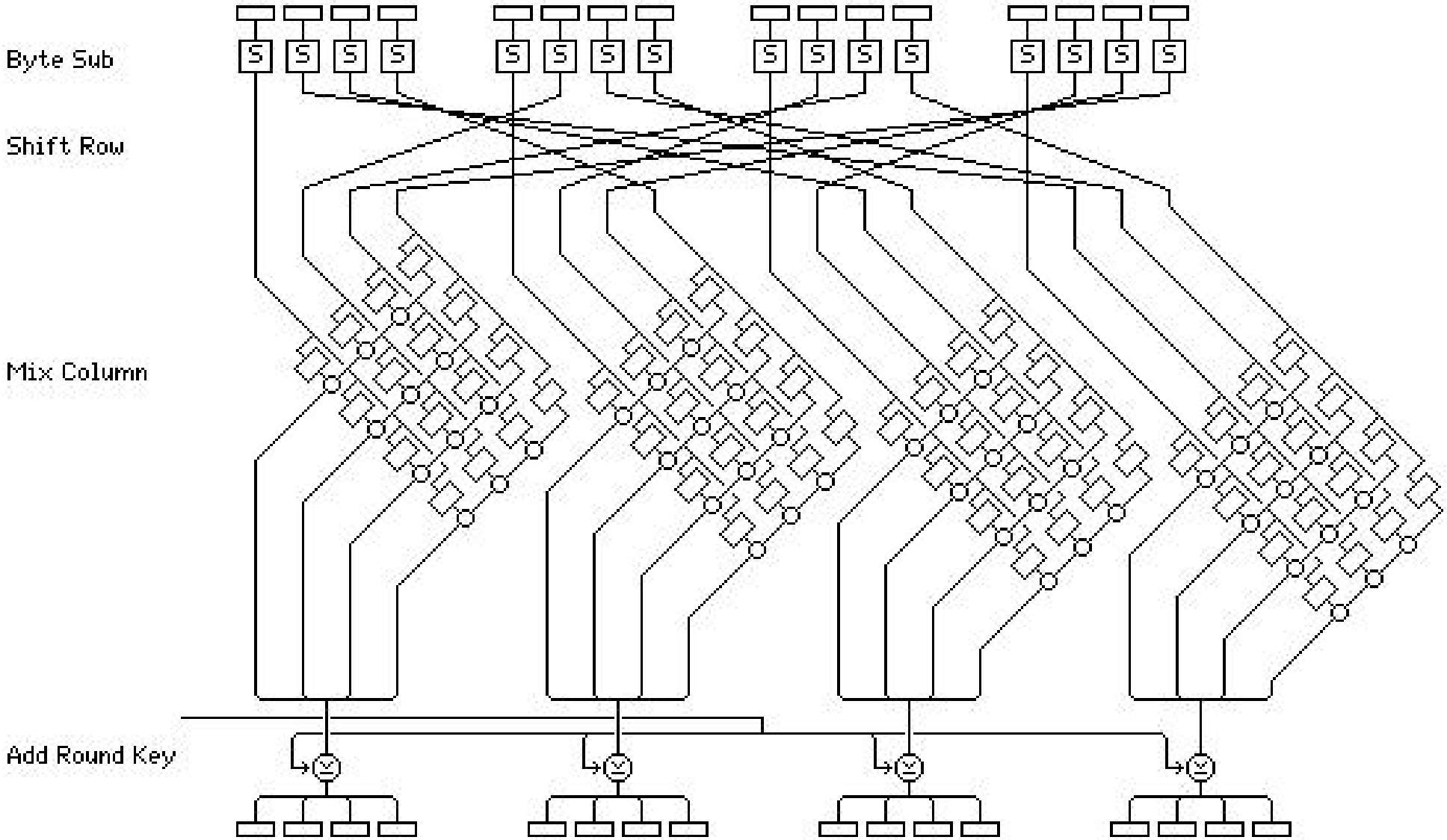
- Byte Sub: jedes Byte im 128 Bits Block wird ersetzt durch einen in einer S-Box vordefinierten Wert ersetzt. Die S-Box entsteht durch ein Ersetzen eines Byte durch dessen Inverses im Galoisfeld GF(2^8), anschließender bitweiser Matrixmultiplikation modulo 2 und abschließendem XOR mit Hexadezimal 63. Diese Operation weist hohe Nicht-Linearität auf und wird als Lookuptable implementiert.

- Shift Row: betrachtet man die Bytes in Matrixschreibweise angeordnet wobei die Spalten zukzessive befüllt werden, so werden die Zeilen dieser Matrix zirkulär nach links geshiftet (0 - 4 Positionen).

- Mix Column: Jede Spalte der resultierenden Matrix wird mit einer speziellen Matrix multipliziert. Diese Multiplikation wird wiederum im GF(2^8) ausgeführt, d.h. die involvierten Bytes werden als Polynome behandelt und nicht als Zahlen. Hat das Ergebnis mehr als 8 Bits, wird mit dem erzeugenden Polynom von GF(2^8) XORed bis 8 Bits erreicht sind. Die Matrixmultiplikation kann sehr effizient auf Bitebene durch XOR ausgeführt werden. Diese Operation, gemeinsam mit Shift Row, führt zu hoher Diffusion.

- Add Round Key: ein XOR mit dem für die aktuelle Runde gültigem Key.

AES Überblick



RSA I

Das RSA-Verfahren beruht auf der Schwierigkeit der Faktorisierung großer Zahlen. Der Public und der Private Key sind Funktionen eines Paares großer (100 oder 200 Stellen) Primzahlen.

- Um die Schlüssel zu erzeugen, wählen Sie zwei zufällig generierte Primzahlen p und q (möglichst gleich groß).
- Berechnen Sie das Produkt $n = pq$.
- Wählen Sie zufällig den Verschlüsselungs Key e, sodaß e und $(p-1)(q-1)$ relativ prim sind.
- Berechnen sie den Entschlüsslungs Key d, sodaß
  $ed \equiv 1 \pmod{(p-1)(q-1)}$
  (d ist also invers zu e.)

RSA II

e und n sind der Public Key, d ist der Private Key. p und q werden nicht mehr benötigt. Um eine Nachricht m zu verschlüsseln wird m in mehrere numerische Blöcke $m_i < n$ aufgeteilt.

Verschlüsselung: $c_i = m_i^e \mod n$
Entschlüsselung: $m_i = c_i^d \mod n$

RSA Geschwindigkeit und Sicherheit

Das RSA-Verfahren ist um einiges langsamer als DES: als Hardware Implementierung etwa 1000 mal, als Software etwa 100 mal. Wichtig ist eine gute Wahl von e. Sehr oft wird 3, 7, $65537(2^{16}+1)$ genommen. Zahlen mit wenig 1 in der binären Darstellung erlauben effiziente Berechnung.

Die Sicherheit von RSA liegt in der Faktorisierung von n. Es ist allerdings nie bewiesen worden, daß dies die einzige Möglichkeit ist, um die Nachricht m aus c und e zu berechnen.

$$ed \equiv 1 \pmod{(p-1)(q-1)}$$

Möglichkeiten der Attacke bestehen etwa in einem Rateversuch von $(p-1)(q-1)$ oder im Brute Force Angriff auf d. Beide Möglichkeiten sind allerdings nicht effizienter als eine Faktorisierung.

One-Way Hash Funktionen I

Die Aufgabe einer One-Way Hash Funktion ist es, einen Fingerabdruck einer Nachricht zu liefern. Die Funktion $H(M)$ operiert auf einer beliebig langen Nachricht M und produziert einen Wert h mit fixer Länge m.

$$h = H(M)$$

Die Grundlage von Hash Funktionen ist daher die Kompression. Eine lange Nachricht M wird in einen kleineren Output h überführt. Zusätzlich sollen die folgenden Anforderungen erfüllt sein:

1. Bei gegebenem M soll h leicht zu berechnen sein.

2. Bei gegebenem h soll es schwierig sein M zu berechnen, sodaß $H(M) = h$ ist.

3. Bei gegebenem M soll es schwierig sein, eine andere Nachricht M' zu finden, sodaß $H(M) = H(M')$ ist.

One-Way Hash Funktionen II

Wenn Mallory die Punkte (2) und (3) durchführen kann, ist H nicht genügend sicher und Betrug leicht. Wenn, z.B. Alice eine Nachricht M digital unterschrieben hat, indem sie die Hash Funktion H auf M anwendet: $H(M)$, könnte Mallory M' erzeugen mit $H(M) = H(M')$. Somit könnte Mallory behaupten, Alice hätte ihm M' geschickt, was natürlich nicht der Wahrheit entspricht.

Für manche Anwendungen sind die geforderten Eigenschaften der One-Way Hash Funktion nicht ausreichend, da auch “Collision Resistence” benötigt wird: Eine Funktion H ist Collision resistent, wenn es schwierig ist zwei zufällige Nachrichten M und M' zu finden, sodaß $H(M) = H(M')$ ist.

Diese Eigenschaft ist wichtig bei der sognannten “Geburtstags Attacke” wichtig.

Die Hash-Funktion MD-5

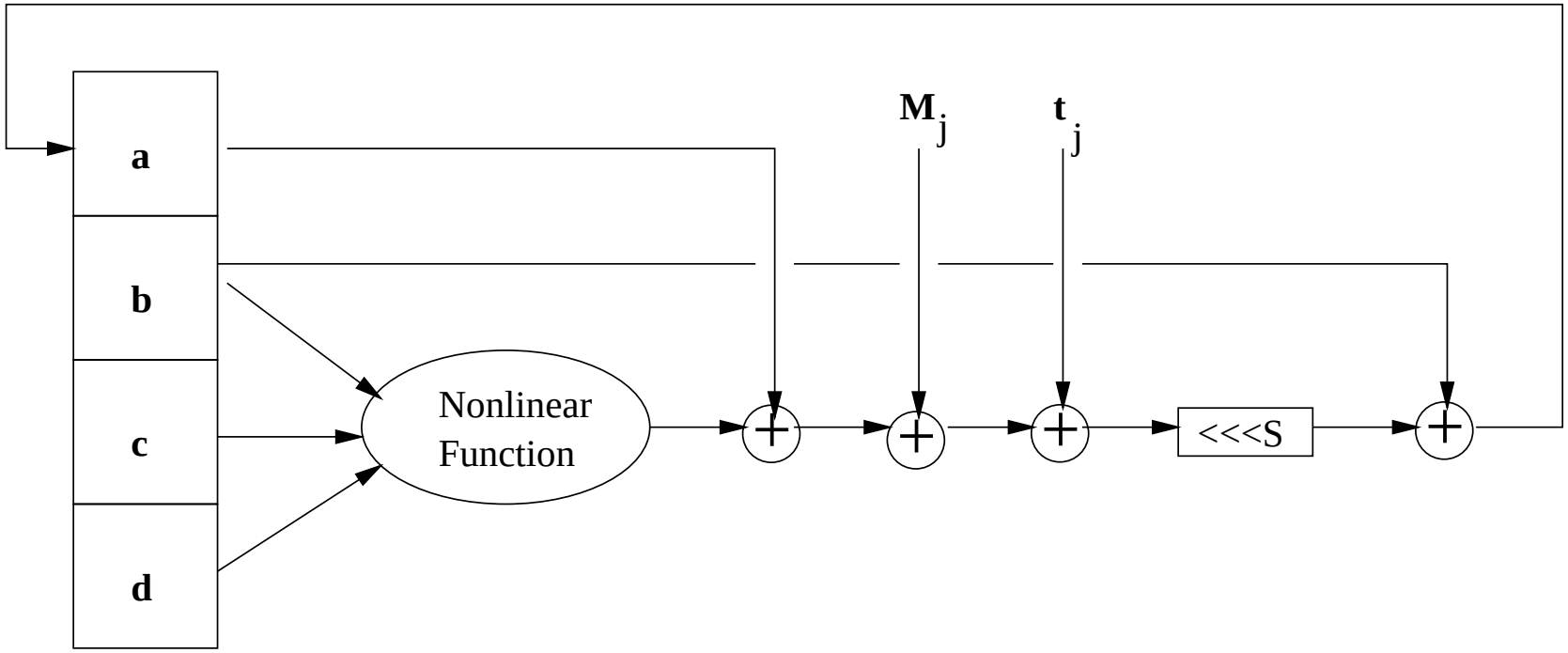
Die Nachricht M wird in 512-Bit Blöcke aufgeteilt, gruppiert in 16 32-Bit Unterblöcken. Der Output besteht aus 4 32-Bit Blöcken, vereinigt in 128-Bit Hash-Werten.

Die Nachrichten werden durch ein Padding auf einheitliche Länge gebracht. Die Anzahl der Bits beträgt ein Vielfaches von 512 minus 64 ($\sharp$ Bits $=512x-64$). Die ursprüngliche Länge der Nachricht wird mit einer 64-Bit Zahl kodiert und an die verlängerte Nachricht angehängt. Somit ist die Länge des Ergebnisses ein exaktes Vielfaches von 512.

Es werden vier 32-Bit Initialvariablen A, B, C, D (chaining variables) verwendet. Die Hauptschleife besteht aus vier Durchläufen. In jedem Durchlauf wird eine bestimmte Operation 16 mal verwendet, wobei in jedem Durchlauf eine andere Operation verwendet wird.

MD-5 (cont.)

Jede dieser Operationen wendet eine nicht lineare Funktion auf drei der Hilfsvariablen a, b, c, d an, addiert das Ergebnis zur vierten Variablen, zu einem Textblock M_j und einer Konstanten t_j. Das Ergebnis wird nach links geshiftet (variabel!) und zu einer der Variablen addiert. Das Ergebnis ersetzt dann a, b, c oder d, die Hilfsvariablen werden dann zu den chaining Variablen addiert.



MD-5 (cont.)

Für jeden Schleifendurchlauf gibt es eine eigene nicht lineare Funktion F, G, H und I.

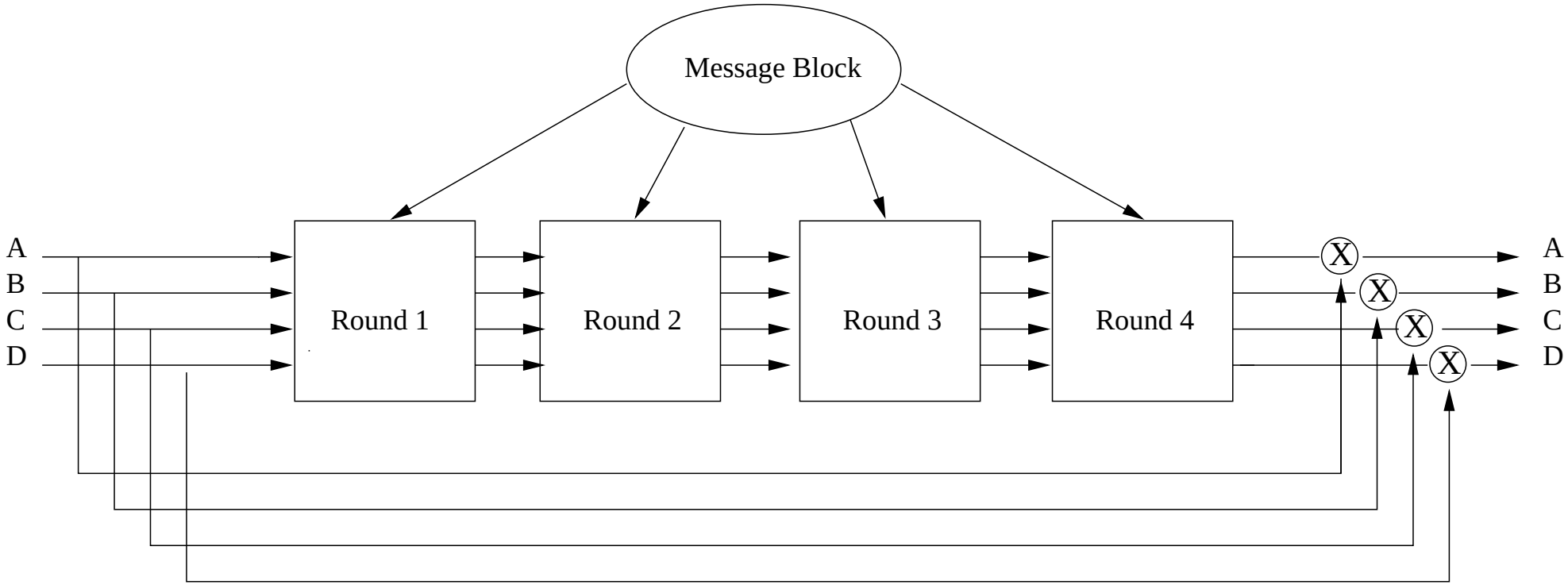
$$\begin{aligned} F(X,Y,Z) &= (X \wedge Y) \vee ((\neg X) \wedge Z) \\ G(X,Y,Z) &= (X \wedge Y) \vee (Y \wedge (\neg Z)) \\ H(X,Y,Z) &= X \oplus Y \oplus Z \\ I(X,Y,Z) &= Y \oplus (X \vee (\neg Z)) \end{aligned}$$

M_j ist der j-te Nachrichtenblock (0-15) und $<<<s$ ist eine linear zirkulärer Shiftoperation. $FF(a,b,c,d,M_j,s,t_i)$ bedeutet dann

$$a = b \oplus ((a \oplus F(b,c,d) \oplus M_j \oplus t_i)^{<<<s}).$$

GG, HH, II sind analog definiert.
$t_i = \text{int}(2^{32}|\sin(i)|)$ in Schleife i, i in radians.

MD-5 (cont.)



Digital Signature Algorithm (DSA) I

1991 wurde der **D**igital **Signature** **A**lgorithmus als Signatur Standard vorgeschlagen. Dies führte zu einem Sturm der Entrüstung in der Industrie, das diese auf Grund der der Verbreitung von RSA einen RSA basierten Standard bevorzugt hätte.

Bei der Entwicklung von DSA war so wie auch bei RSA die NSA beteiligt. Das Standardisierungs- verfahren wurde ohne öffentliche Ausschreibung durchgeführt. Diese Vorgehensweise schürte natürlich das Mißtrauen diverser Institutionen.

p ist Primzahl mit L-Bits, $512 \leq L \leq 1024$, $L = K64$.
q ist ein 160-Bit Primfaktor von $p-1$
$g = h^{\frac{p-1}{q}} \bmod p > 1$, mit $h < p-1$.
wähle $x < q$ und berechne
$y = g^x \bmod p$.

DSA ist ein dem El Gamal Algorithmus sehr ähnliches Verfahren. Es verwendet eine One-Way Hash Funktion $H(M)$, den sogenannten Secure Hash Algorithmus. p, q und g sind öffentlich. y ist der Public Key. x ist der Private Key.

Digital Signature Algorithm (DSA) II

Das zughörige Protokoll sieht folgendermaßen aus:

1. Alice generiert ein zufälliges $K < q$.

2. Alice generiert
   $r = (g^K \bmod p) \bmod q$
   $s = (K^{-1}(H(M) + xr)) \bmod q,$
   wobei r und s ihre Unterschrift darstellen. Sie schickt beides an Bob.

3. Bob berechnet
   $$\begin{aligned} w &= s^{-1} \bmod q \\ u_1 &= (H(M)w) \bmod q \\ u_2 &= (rw) \bmod q \\ v &= ((g^{u_1} y^{u_2}) \bmod p) \bmod q \end{aligned}$$
   Wenn v gleich r ist, ist die Unterschrift verifiziert.

Da r nicht von der Nachricht abhängt, können K, K^{-1} und r vorberechnet werden, um das tatsächliche Signieren zu beschleunigen. Der Standard enthält weiters eine Methode, die Primzahlen zu generieren. Das schließt die Verwendung gefährlicher Moduli aus.

Netzwerksicherheit

Grundsätzlich können Sicherheitsfeatures auf unterschiedlichen Ebenen realisiert werden.

- Applikationsebene: Vorteil ist, daß nur die Applikationen sicher sind die diese Eigenschaft auch wirklich brauchen (kein Overhead), andererseits muß sich jede Applikation bzw. auch jeder Nutzer um die Sicherheit kümmern (z.B. PGP, ssh, Kerberos, shttp, ...).

- Netzwerkebene: Vorteil ist die völlige Transparenz, Nachteil der enorme Overhead da unabhängig von der Applikation Sicherheitsfeatures eingesetzt werden (IPSec, DNSSec).

- Linkebene: v.a. im militärischen Bereich verwendet, hier ist jeder Netzwerklink mit Kryptographischer Hardware versehen. Die Sicherheit ist exzellent, solange hinter den Links “sicheres” Gebiet ist. Im Internet natürlich nicht einsetzbar.

Sicherheit des IP Protokolls: IPSec

Die Verbesserung des IP-Protokolls bezglich einer Verschlsselung und Authentifizierung ist bereits seit mehreren Jahren im Gesprch. Fr Version 4 (also das derzeit aktuelle Protokoll) arbeitet die IETF-Arbeitsgruppe ”Internet Protocol Security Protocol (ipsec)” bereits seit 1993 an einer Lsung. Fr Version 6 (IPng oder IPv6, die Nachfolgeversion) wurden diese Sicherheitsmechanismen bereits von Anfang an gefordert. Diese Mechanismen werden als IPSec bezeichnet. Anwendungsgebiete sind z.b.:

- Sichere Verbindung von Zweigstellen übers Internet
- Sicherer remote Access
- Verbesserung von e-commerce Sicherheit
- .....

IPSec Anwendung

Klassisches Szenario von IPSec Benutzung ist eine Organisation, die mehrere LANs an verschiedenen Orten betreibt. Nicht-abgesicherter IP Verkehr wird innerhalb der LANs eingesetzt, zur Kommunikation zwischen den LANs über ein WAN oder Internet werden IPSec Protokolle verwendet, die in den entsprechenden Routern oder Firewalls laufen.

Der große Vorteil von Sicherheit die über IPSec bereitgestellt wird ist daß diese Art von Sicherheit für den User völlig transparent ist, da sie unterhalb des Transport Layer (TCP,UDP) angesiedelt ist. Keinerlei Anwendungssoftware muß angepasst werden. Nachteil ist die höhere Verarbeitungskomplexität für jede Kommunikation (also auch für Urlaubsgrüße).

IPSec AH & ESP

Sowohl für IPv4 und IPv6 sind die Sicherheitsfeatures als Extension Headers hinter dem Haupt IP Header implementiert, es gibt zwei Varianten:

- Authentication Header (AH): hier werden die Daten und die invarianten Felder des äusseren IP Headers mit einem MAC - wegen der Geschwindigkeit statt einer Signatur - authentifiziert.

- Encapsulating Security Payload (ESP): kombiniertes Verschlüsselungs/Authentifizierungs bei ESP Authentifizierung wird der Header nicht authentifiziert sondern nur die Payload.

Der Standard Verschlüsselungsalgorithmus ist DES in CBC Modus, aber auch IDEA, triple DES und weitere Algorithmen können verwendet werden. Zur Authentifizierung werden MD5 und SHA1 verwendet.

IPSec Modi & SA

Die beiden Modi Transport und Tunnel unterscheiden sich bezüglich der Anwendungsszenarien: während der Transport Mode v.a. für sichere End-to-End Kommunikation verwendet wird, den Header unverändert lässt und die IP Payload schützt, wird beim Tunnel Mode das gesamte IP Paket geschützt und mit einem neuen IP Haeder versehen und für das Tunnelling zwischen Routern verwendet.

Ein wesentliches Konzept ist das der Security Association (SA) – das ist eine Einwegbeziehung zwischen Sender und Empfänger und wird durch die IP-Empfangsadresse und den Security Parameter Index (SPI) festgelegt. In jeder IPSec Implementierung gibt es eine Datenbank, die die Parameter festlegt, die zu einer SA gehören (AH und ESP Information bezüglich Algorithmen, Schlüssel, Gültigkeitsbereich von Schlüsseln, IPSec Protocol Mode - Tunnel vs. Transport u.s.w.).

IPSec: IKE

IPSec nimmt an, daß bereits eine SA etabliert worden ist stellt aber keinen Mechanismus zur Verfügung eine SA zu kreieren. IPSec führt nur die paketorientierte Verarbeitung durch. Für spezielle Anwendungsszenarien können verschiedene SAs ineinander geschachtelt werden, das Key Management verwendet eine Verbesserung des Diffie-Hellman Key Exchange, das auch ellipische Systeme zuläßt.

Das Etablieren der SAs übernimmt IKE (Internet Key Exchange), das einen authentifizierten, sicheren Tunnel schafft und SAs zwischen den Teilnehmern aushandelt.

- Authentifizierung:
  - ⋆ Pre-shared keys: identische Keys sind auf den Hosts vorinstalliert; IKE Authentifizierung geschieht durch Berechnung und Senden eines verschlüsselten Hash Wertes von Daten die den vorinstallierten Key enthalten, wenn beide das gleiche generieren können, müssen beide im Besitz des gleichen Keys sein.
  - ⋆ Public-Key Methode: jeder Teilnehmer generiert Zufallszahl und verschlüsselt diese mit dem public Key des Anderen (RSA). Jeder Teilnehmer generiert dann mit seinem private Key einen verschlüsselten Hash der Zufallszahl des Anderen und sendet diese. Kann auch direkt mit digitalen Signaturen gemacht werden (DSS).

Kritik an IPSec

IPSec wird stark kritisiert. Die hohe Komplexität des Verfahrens, Resultat des Entstehungsprozesses (ein Kommittee wählte nicht aus Submissions aus sondern bastelte IPSec selbst was viele Kompromisse notwendig machte), bedingt eine hohe Anfälligkeit gegenüber Fehlkonfiguration. Neben der Kritik an der “Unlesbarkeit” der Dokumentation wird eine Elimination des Transport Modes und des AH vorgeschlagen (der Overhead im Tunnel Mode könnte leicht durch einen speziellen Headerflag ausgeglichen werden, der angibt, wenn beide IP Header gleich sind und ESP müsste nur um Header Authentifizierung erweitert werden um volle AH Funktionalität zu erreichen).

Authentifizierung im DNS: DNSSec

Das Domain-Name-System kommt immer dann zum Einsatz, wenn IP-Adressen in Rechnernamen umgesetzt werden müssen oder eine umgekehrte Umsetzung erforderlich ist. Durch einen Angriff kann die Adreßumsetzung für ganze Domains übernommen werden, so daß weitreichender Einflu auf die Funktion des gesamten Netzes genommen werden kann. Mit Hilfe solcher Manipulationen können Pakete auch um- und fehlgeleitet werden, so da sich ein Angreifer in den Besitz der übertragenen Informationen bringen können oder sich sogar als fremde Rechner ausgeben können. In diesem Bereich sind bereits viele Angriffe bekannt geworden (DNS Spoofing) und die Schwachstellen wurden auch dokumentiert.

Die grundlegenden Probleme sind jedoch ohne eine Änderung des gesamten DNS nicht lösbar da das System auf der Öffentlichkeit der Daten basiert und auf das korrekte Arbeiten aller Teilnehmer vertraut. Auch zum Schutz des DNS werden darum kryptographische Methoden, die auf Public-Keys basieren, entwickelt. Ziel ist jedoch im Gegensatz zu IPSec nicht Verschlüsselung sondern die Gewährleistung der Integrität (keine Veränderung) und Authentizität (Daten stammen wirklich aus der vorgegebenen Quelle) der DNS Daten.

Main Services von DNSSec

- Key distribution: erlaubt nicht nur das Beziehen des public Key eines DNS Namens um die Authenzität der DNS Zonen Daten zu prüfen sondern mit diesem Schema können grundsätzlich alle Keys die mit einem DNS Namen verbunden sind administriert werden, auch für ganz andere Zwecke.

- Data origin authentication: das Resource Record (RR) Set einer DNS Zone wird kryptographisch signiert (verschlüsselter Hash des RRSet). Der Hash wird mit dem private Key des primären DNS Server der Zone verschlüsselt. Verifikation geschieht einerseits durch Hashen des RRSet und andererseits durch Entschlüsselung des verschlüsselten Hash mit dem public Key des primären DNS Servers der Zone (der mit Hilfe der Key distribution Funktionalität verfügbar ist).

- DNS Transaction and Request Authentifizierung: hier geht es darum ob eine Antwort eines DNS Servers tatsächlich die ursprüngliche Frage beantwortet und ob die Antwort von dem Server kam der auch befragt wurde. Das wird durch eine Signatur einer Concatenation von Frage und Anwort erreicht die alle notwendigen Daten für Verifikation enthält. Zusätzlich gibt es Mechanismen für starke Authentifizierung des dynamischen Updates sekundärer DNS Server.

Neue RRs

Um diese Services realisieren zu können, wurden neue RRs definiert:

- Key RR: hier ist der (die) public Key(s) für einen DNS Namen abgelegt. Zusätzlich wird der zugehörige Algorithmus (z.B. RSA, MD5, Diffie-Hellman, DSA, elliptic curve) und der Protokolltyp (TLS, email, DNSSEC, IPSec) definiert. Nicht inkludiert sind Key Zertifikate, die im CERT RR abgelegt sind.

- SIG RR: enthält die Signatur des RRSet und die notwendigen algorithmischen Informationen sowie den Gültigkeitszeitraum der Signatur.

- NXT RR: dient zur Authentifizierung nicht existierender RRSets, falls ein DNS Name nicht existiert oder der RR Typ in einer Anfrage für einen Namen nicht existiert. Dies wird durch eine vordefinierte Anordnung der RRSets erreicht.

PKI für DNSSec

In Security Aware DNS-Servern gibt es zwei Formen von Authentifizierung:

- Off-line: der private Key der Zone darf keinesfalls kompromittiert werden, er ist daher nicht on-line verfügbar und wird jedesmal zur Erstellung der SIG RR generiert/retrieved.

- On-line: Authentifizierung von DNS Transaktionen geschieht häufiger - dies wird mit anderen Keys, den Server-Keys gemacht, welche daher on-line verfügbar sein müssen.

Es bleibt die Frage, wie die die public Keys verteilt werden. Wird DNS selbst durch Abfrage des key RR verwendet stellt sich die Frage wie die Authenzität der public Keys im Key RR gewährleistet wird. Diese müssen dann von der Masterzone signiert sein, wobei die Frage entsteht, wie die Authentizität deren public Keys gewährleistet ist. Man braucht daher irgendwann eine “trusted zone”. Statische Konfiguration der Keys skaliert wiederum nicht gut - wird ein Key geändert, müssen alle Konfigurationen geändert werden.

Verschlüsselung von Visuellen Daten

- Grundlagen
- Pay TV Systeme,
- DVD,
- BLU RAY,
- Partielle Verschlüsselung: Bilder,
- Partielle Verschlüsselung: Videos,
- Spezielle Methoden.

Grundlagen

Der klassische Kryptograph sagt: “Egal welche Daten vorliegen, man nehmen einen sicheren Cipher und verschlüssle diese Daten. Dazu gibt es keine Alternativen”.

Gibt es irgendwelche Gründe die dafürsprechen multimediale Daten (hier speziell visuelle Daten) anders zu behandeln als “klassische” Daten die verschlüsselt werden ????

- Verschiedene Funktionalitäten / Ziele die durch Verschlüsselung erreicht werden sollen.
- Extreme Datenmengen, eventuelle Echtzeitanforderungen
- Unterschiedliche Anforderungen an das Sicherheitsniveau (e.g. VoD, Videoconferencing, TV-News Broadcast, Medizinische Bilddaten) und verschiedener “Wert” der Daten (wenn die Kosten für das Brechen der Verschlüsselung höher sind als der Wert/Preis der Daten, wird man die Daten kaufen)
- Unterschiedlich “wichtige” Angreifer
- QoS in Netzwerken die zu Rate Adaptation bzw. Transcoding führen, Streaming, Übertragungsfehler

Offenbar genug um sich das genauer anzusehen !!!!

Example: Medical Applications

The organisation of todays health systems often suffers from the fact that different doctors do not have access to each others patient data. The enormous waste of resources for multiple examinations, analyses, and medical check-ups is an immediate consequence. In particular, multiple acquisition of almost identical medical image data and loss of former data of this type has to be avoided to save resources and to provide a time-contiguous medical report for each patient. A solution to these problems is to create a distributed database infrastructure where each doctor has electronic access to all existing medical data related to a patient, in particular to all medical image data acquired over the years. Classical Telemedicine adds interactivity to that.

There is urgent need to provide and protect the confidentiality of patient related medical image data when stored in databases and transmitted over networks of any kind.

Example: Video Conferencing

In todays communication systems often visual data is involved in order to augment the more traditional purely audio-based systems. Whereas video conferencing (VC) has been around to serve such purposes for quite a while and is conducted on personal computers over computer networks, video telephony is a technology that has been emerging quite recently in the area of mobile cell phone technology.

No matter which technology supports this kind of communication application, the range of possible content exchanged is very wide and may include personal communication among friends to chat about recent developments in their respective relationships as well as video conferences between companies to discuss their brand-new product placement strategies for the next three years. In any case, each scenario requires the content to be protected from potential eavesdroppers for obvious reasons.

Example: Surveillance

The necessary protection of public life from terroristic or criminal acts has caused a tremendous increase of surveillance systems which mostly record and store visual data. Among numerous applications, consider the surveillance of public spaces (like airports or railway stations) and casino-gambling halls. Whereas in the first case the aim is to identify suspicious criminal persons and/or acts, the second application aims at identifying gamblers who try to cheat or are no longer allowed to gamble in that specific casino.

In both cases, the information recorded may contain critical private informations of the persons recorded and need to be protected from unauthorised viewers in order to maintain basic citizens' rights. This has to be accomplished during two stages of the surveillance application: first, during transmission from the cameras to the recording site (e.g. over a firewire or even wireless link), and second when recording the data onto the storage media.

Visual Example: Privacy & Surveillance

(a) permutation

(b) sign bit encryption

Abbildung 1: Visuelle Beispiele für selektives Schützen personenbezogener Information.

Example: Video on Demand

VOD is an entertainment application where movies are transmitted from a VOD server to a client after this has been requested by the client, usually video cassette recorder (VCR) functionalities like fast forward or fast backward are assumed (or provided) additionally. The clients' terminals to view the transmitted material may be very heterogeneous in terms of hardware capabilities and network links ranging from a video cell phone to a HDTV station connected to a high speed fibre network.

To secure the revenue for the investments of the VOD company, the transmitted movies have to be secured during transmission in order to protect them from non-paying eavesdropping "clients" (encryption), and additionally, some means are required to disable a legitimate client to pass over the movies to a non-paying friend or, even worse, to record the movies, burn them onto DVD and sell these products in large quantities (watermarking and fingerprinting). Similar challenges have to be met with the DVD system ;-)

Consider a heterogeneous network with varying bandwidth, with transitions between wired and wireless. How to facilitate e.g. rate adaptation without decryption, transcoding, and re-encryption ?

Example: Pay-TV News

Free-TV is financed via commercials (everywhere) and/or via governmentally imposed, tax-like payments (like e.g. in Austria where everybody who owns a TV-set has to pay those fees no matter if he watches federal TV channels or not). Contrasting to that, Pay-TV is financed by the subscription payments of the clients. As a consequence, only clients having payed their subscription fees should be able to consume Pay-TV channels. This is usually accomplished by encryption of the broadcasted content and decryption in the clients' set-top box, involving some sort of smartcard technology.

Whereas the same considerations apply as in the case of VOD with respect to protecting the content during transmission, there is hardly any threat with respect to reselling news content to any other parties since news data loose their value very quickly.

Application-driven Media Encryption Security Levels

- Cryptographic Security: no information about the plaintext shall be deductible from the ciphertext. E.g., this includes indistguishability under a chosen plaintext attack (IND-CPA): Given two plaintexts and a corresponding ciphertext, an adversary cannot identify the correct plaintext with probability better than 0.5.

- Content security/Confidentiality: Information about the plaintext may leak, but the video content must not be descernible.

- Sufficient encryption: The content must not be consumable with pleasant viewing experience due to high distortion, but it is acceptable that content is descernible.

- Transparent/perceptual encryption: A preview image needs to be decodeable, but a higher quality version has to be protected.

Which applications would you assign to these security levels ? This depends on which aims you want to achieve and whom you want to protect e.g. VoD: content provider revenues vs. customer privacy !

Example: Request for Cryptographic Security in VoD

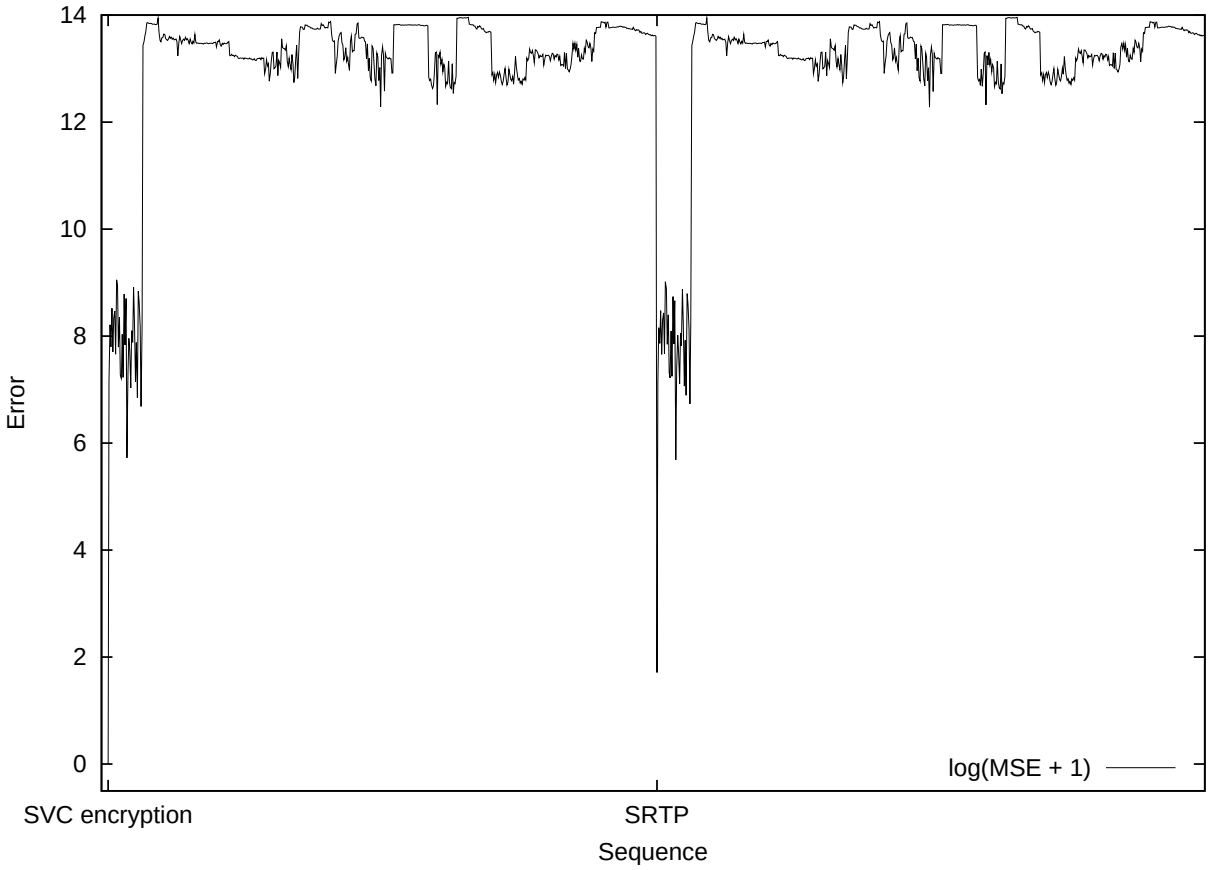
We have split well-known CIF sequences (Akiyo, Bus, City, Coastguard, Container, Crew, Flower, Football, Foreman, Harbour, Ice, Mobile, News, Silent, Soccer, Tempete, Waterfall) into non-overlapping subsequences of 8 frames. This results in 582 distinct sequences, of which some are very similar, e.g., the subsequences of the Akiyo sequence. The bitstreams were generated using the Joint Scalable Video Model (JSVM) [**?**] 9.14 software. The scalable bitstream contains a QCIF substream (compliant to the H.264 baseline profile) and two CIF MGS layers to enable bitrate adaptation.

In order to assess the similarity between packet traces, the mean squared error of the packet lengths is considered. If the number of packets differs between two packet traces, the MSE is calculated for the smaller number of packets. Thus, the difference between two packet traces pt_1 and pt_2 is defined in the following way:

$$\mathrm{d}(pt_1, pt_2) = \sum_{i=1}^{min(n_1,n_2)} (l_{1i} - l_{2i})^2$$

The overall number of packets for pt_1 is n_1 and the corresponding packet lengths are denoted by l_{1i} and similarly for pt_2. Two different encryption modes are considered (SVC-specific and the SRTP encryption).

Surprising Result



Despite very similar subsequences, each packet trace was unique for both the SVC-specific and the SRTP encryption. This fingerprint is not just unique, but also a weak measure for sequence similarity (shows MSE between the packet trace of the first subsequence from the Akiyo sequence with SVC-specific encryption and all other subsequences, even those encrypted with SRTP).

Implementation Levels of Media Encryption

- File Level: Encryption is applied regardless of the file type and structure.
- Transport Level: Encryption is applied regardless of the content – packets of stream segments of the transport layer are encrypted (e.g. using IPSec, TLS, SRTP).
- Metaformat Level: Encryption is applied within the scope of a metaformat, such as the ISO base media file format. Approaches which employ bitstream descriptions (e.g. MPEG-21 gBSD) and encryption fall into this category.
- Codec format Level: Encryption directly applied at the bitstream level, usually applied to preserve codec-specific features like scalability.

Kerkhoff's Principle: Is also valid for Media Encryption ! Security of a Cryptosystem must not stem from the secrecy of the technology but from secret cryptographic key material !!! If algorithms are kept secret, the system is not secure and higly endangered.

Pay TV Systeme: Typen

- Analoge Systeme: kein kryptographischer Schutz, Teile der Information werden nicht standardgemäß gesendet (z.B. vertauschte Synchronisationsinformationen). Dadurch kann das Decodieren nur von einem Decoder des Auslieferers vorgenommen werden. UNSICHER. Wird heute ausschließlich von KabelTV Betreibern verwendet.

- Hybride Systeme: Das Signal wird entsprechend einem analogen TV Standard gesendet (z.B. PAL, SECAM, NTSC). Dieses Signal wird in einem Framebuffer digital verschlüsselt. Dazu wird ein Decoder benötigt, der den verwendeten Algorithmus in Form von Hardwarekomponenten (meist Smartcards) besitzt. Beispiele: VideoCrypt, Syster Nagravision, EuroCrypt

- Digitale Systeme: Ursprünglich digitales Signal (meist MPEG-2) wird moduliert und analog übertragen. Verschlüsselung gleich wie bei hybriden Systemen. Beispiel: DVB

Pay TV Systeme: Gebührenentrichtung

- Pay per Channel
- Prebooked pay-per-view: bestimmte Filme oder Zeiten werden vorgebucht.
- Impulse pay-per-view: Chipkartenkonto auf dem Guthaben in vorher bezahlter Höhe ist.
- Near VoD: Auswahl innerhalb eines Zeitintervalls

Pay TV Systeme: Sicherheitskonzept I

Verschlüsselung ist das primäre Sicherheitskonzept. Die Entschlüsselungssysteme werden beiden Empfängern bereitgestellt (Erwerb oder Miete). Empfangssysteme bestehen aus dem Decoder und dem Sicherheitsmodul.

- Erste Systeme: Sicherheitssystem war im Dekoder integriert. Dies hatte bei Kompromittierung der Schlüssel oder der geheimen Algorithmen den Austausch des Gesamtsystems zur Folge.

- Heutige Systeme: meist Chipkarte als Sicherheitsmodul.

Die beiden wesentlichen Komponenten des Sicherheitskonzeptes sind das Zugriffskontrollsystem und das eigentliche Scrambling System.

Pay TV Systeme: Sicherheitskonzept II

1. Scrambling System: Ver- und Entschlüsselung der Bilddaten durch das vom Zugriffskontrollsystem bereitgestellte Kontrollwort (CW)

2. Zugriffskontrollsystem:

   - Bereitstellung von Kontrollwörtern für das Scrambling System in Echtzeit, die Übermittlung muß vertraulich erfolgen, da jeder, der CW kennt, die Daten entschlüsseln kann.
   - Verwaltung der Berechtigungen der einzelnen Benutzer

   Entsprechend diesen unterschiedlichen Aufgaben gibt es zwei verschiedene Arten von Nachrichten an das Zugriffskontrollsystem.

Pay TV Systeme: Sicherheitskonzept III

1. ECM (Entitelment control message - für globale Informationen): Übertragung neuer CWs zum Entschlüsseln des Programms und Mitteilungen der generellen Programm Bedingungen (z.B. PG Informationen, Gebühren müssen bezahlt sein, ...). Die CWs werden mit dem sog. Systemschlüssel verschlüsselt, der auf der Smartcard implementiert ist. Für alle Benutzer identisch.

2. EMM (Entitlement management message - für individuelle Informationen): zur Änderung der individuellen Berechtigungen zum Empfang von Programmen auf der Empfängerseite. Diese Nachrichten sind natürlich sensibel und müssen geschützt werden (z.B. durch das Fiat-Schamir Identifikationsverfahren bei VideoCrypt und ein public-key Verfahren bei EuroCrypt).

Pay TV Systeme: Sicherheitskonzept IV

Wie können nun Decoder gesperrt werden, wenn die Gebühren nicht bezahlt wurden ?

1. Negativ Adressierung: Löschen von Berechtigungen mit EMM, d.h. das Sicherehitsmodul wird teilweise oder vollständig deaktiviert (bei Piratenkarten ist diese Ausschaltfunktion nicht implementiert !)

2. Positiv Adressierung: es werden falsche/ungültige Schlüssel an die entsprechenden Adressen geschickt. Das bietet auch Sicherheit gegen Piratenkarten, da auch sie wertlose Schlüssel erhalten. Problem ist allerdings die Verteilung der wertlosen Schlüssel (zu viele EMMs).

Angriffe auf PayTV Systeme

- Piratensysteme: Manipulierte Decoder oder Smartcards, rechtlich problematisch (Funktionsprinzip der Hardware papentrechtlich geschützt), jedoch uneinheitliche Rechtslage und -auslegung in Europa

- Angriffe auf das Scrambling System: Brute-force Attacken gegen Permutationen (dazu später). Keine Patenrechtsverletzung und kaum Gegenmaßnahmen seitens des Betreibers möglich.

- Aufzeichnung und Weitergabe von Schlüsseln: da die CWs für alle Smartcards identisch sind und identisch verschüsselt sind, können sie aufgezeichnet werden. Ebenso kann ein verschlüsseltes Signal mit einem VCR augenommen werden und später mit den aufgezeichneten CWs entschlüsselt werden (“offline internet card sharing”). Ein PC emuliert dabei die Smartcard.

Nagravision/Syster (z.B. Premiere Analog)

Für PAL, SECAM und NTSC

Grundprinzip: Zeilenvertauschung in Blöcken von je 32 Zeilen.

Alle 256 Fields (Haldbilder bei Interlaced Video) wird diese Vertauschung durch neues CW geändert.

Die Zeilenvertauschung (Permutation) wird durch einen Zufallszahlengenerator gesteuert, der eine Liste von 256 Zahlen zwischen 0 und 31 durchgeht.

Das CW ist in diesem Fall der Seed des Generators, der festlegt, wo in der Liste begonnen wird und mit welcher ungeraden Schrittweite die Liste round-robin durchgegangen wird. Das ergibt $256 \times 128 = 32768$ mögliche Permutationen.

Angriff auf Nagravision/Syster I

Vorraussetzung: die Liste der 256 Zahlen ist bekannt. Ist sie es nicht, wird die Attacke wesentlich aufwendiger.

Die 32768 möglichen Permutationen werden berechnet.

Grundprinzip: wenn zwei im unverschlüsselten Bild aueinanderfolgende Zeilen, die im verschlüsselten Bild von einender entfernt liegen, nach einer Transformation mit einer bestimmten Permutation wieder korrekterweise untereinander liegen, dann ist dieser Schlüssel mit hoher Wahrscheinlichkeit der Richtige.

Man benötigt jedoch mehrere Zeilenpaare, um ausreichende Sicherheit zu erhalten.

Vorraussetzung: Zeilen die nahe beisammen sind (im Original) sind ähnlich.

Beurteilungskriterium: z.B. kleine Summe der pixelweisen Helligkeitsunterschiede.

Angriff auf Nagravision/Syster II

Tradeoff zwischen Genauigkeit und Effizient des Angriffs:

- Je mehr Zeilenkandidaten, desto genauer (und desto langsamer)
- Je mehr Pixel pro Zeile betrachtet werden, desto genauer (und desto langsamer)

Dieser Angriff ist in Echtzeit erst in den letzten Jahren möglich, die Aufzeichnung des verschlüsselten Signals und die offline Entschlüsselung ist jedoch schon länger möglich.

Fazit: Der Aufwand für eine erfolgreiche Attacke steht wohl in vernünftiger Relation zum Wert der Daten, da diese Art des Angriffs ohnehin nur für eine kleine Zahl von Eingeweihten möglich ist.

VideoCrypt I + II (z.B. SKY)

Für PAL, SECAM und NTSC, entwickelt von Thomson Consumer Electronics

Jede Framezeile wird an einem geheimen Cut-point in zwei Stücke geteilt und die Stücke vertauscht (cut and rotate)

256 mögliche cut points sind zugelassen, wobei diese 12 - 15 % der Zeilenweite vom Rand weg sein müssen

Einige male pro Sekunde wird eine 32 Byte Nachricht gebroadcastet.

Alle 2.5 Sekunden generiert ein MAC 60 Bit, die als Seed für einen Zufallszahlengenerator verwendet werden, der 8 Bit ausgibt.

Im Dekoder generiert die Smartcard den MAC Output.

Die 32 Byte Nachrichten enthalten auch EMMs und eine Signatur anhand derer die Smartcard die Authentizität prüfen kann.

Angriff auf VideoCrypt

Durch die Ähnlichkeit von benachbarten Zeilen können die 256 cut points durchgetestet werden. Wieder gilt daß die Zuverlässigkeit aber auch die Komplexität bei der Verwendung von mehreren Zeilen steigt.

1993 wurde die Hash Funktion (vermutlich durch Indiskretion beim Entwickler) bekannt. Bis zum Wechsel zu einer neuen Smartcard (dauerte ca. ein Jahr) waren sehr viele Piratenkarten im Umlauf.

Fazit: praktisch identisch zu Syster

EuroCrypt

Frankreich: Kabelnetz Pay-TV, Skandinavien: Satelliten Pay-TV

Durch Verwendung verschiedener Smartcards in einem Dekoder ist der Empfang von mehreren Anbietern mit nur einem Gerät möglich

Verschlüsselung und Sicherheit ähnlich wie bei den beiden vorigen Systemen.

Digital Video Broadcast (DVB)

DVB basiert auf MPEG-2 was die Übertragung von bis zu 30 digitalen Fernsehkanälen auf einem Satellitenkanal erlaubt

DVB Receiver (Set-top Box) besteht aus:

- Kabel/Antennenanschluß
- Receiver und Demodulierung
- Fehlerkorrektur
- Access Control und Entschlüsselung (optional)
- MPEG Demultiplexer
- Dekoder für Video, Audio, Text
- D/A Wandler (PAL, NTSC, ...)

DVB Encryption

Zweigeteiltes Sicherheitskonzept: Eigentliche Verschlüsselung des MPEG Datenstroms (“descrambling”) und die geschützte Übertragung des ECMs und EMMs (“decryption”).

Für das descrambling wurde ein einheitlicher Algorithmus standardisiert, der aus einem 64Bit Blockcipher und einem Streamcipher besteht (common scrambling algorithm). Technische Einzelheiten nur für Hardwarehersteller erhältlich (non-disclosure agreement).

Für die decryption wurde keine gemeinsame Lösung gefunden (Angst vor Piratenkarten)

Um doch Kanäle von verschiedenen Betreibern mit einer Set-top Box empfangen zu können gibt es zwei Möglichkeiten: Common Interface und Simulcrypt.

DVB: Common Interface (CI) und Simulcrypt

- Common Interface (CI): ist ein standardisierter Slot in der Set-top Box für eine oder mehrere PCMCIA Karten (CAMs - conditional access modules). Auf dem CAM befindet sich ev. ein alternativer Scrambling Algorithmus und/oder Software/Schlüssel für die Decription.

- Simulcrypt: Die Broadcaster können die Daten für das Decryption System passend für mehrere Systeme senden. Funktioniert dann, wenn der common scrambling algorithm eingesetzt wird und wenn der sich der Set-top Box Verkäufer (meist ein Broadcaster) und der Broadcaster von dem man was sehen will, einig sind.

DVD: Sicherheitskonzept I

Das DVD Sicherheitskonzept beruht auf trusted Hardware, d.h. man geht davon aus daß die digitalen Daten den geschützten Bereich der lizensierten Hardware nicht verlassen. Diese Hardware hält sich natürlich an die Regeln.

- 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. 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 ver”andert werden, dies ist allerdings nur einige Male möglich. Eigentlich obsolet.

DVD: Sicherheitskonzept II

- APS (Macrovision): Dieser analoge Kopierschutz kann vom Produzenten auf der DVD-Disc durch “trigger bits” f”ur 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 nat”urlich 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 verhindern soll.

DVD: Sicherheitskonzept III

- CGMS cont.: In den Video-Daten werden die CGMS-Informationen 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 !

DVD: Sicherheitskonzept IV

- 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 Schl”ussel 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. DTCP v.a. für IEEE 1394 (Firewire) entwickelt.

DVD: Sicherheitskonzept V

- Content Scrambling System (CSS): Die Daten auf der DVD werden durch CSS verschlüsselt (geheimes Verfahren) 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 entschlüsselten 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 verschl”usselte DVD-Disc 1:1 zu kopieren. Es hindert also den Normal-User eine Kopie zu erzeugen, nicht jedoch professionelle Kopierer.

Der DVD/CSS Hack

“We found that one of the companies had not encrypted their CSS decryption code, which made it very easy for us”.

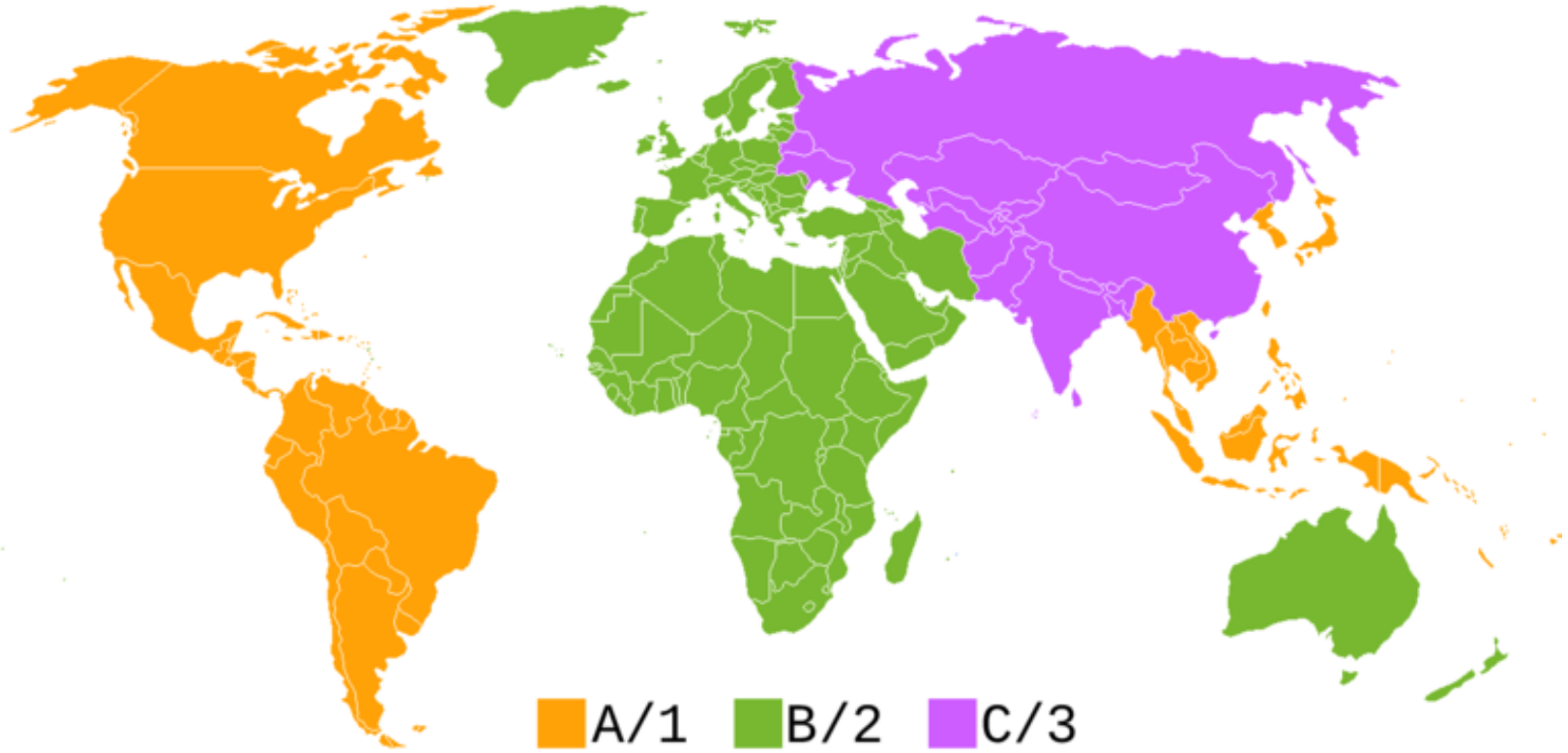
Jeder DVD Player hat seinen eigenen Unlock-Schlüssel (sollte verschlüsselt sein !) und jede DVD Disc hat 400 verschiedene verschlüsselte 5-byte CSS Schl”ussel. So kann jeder Schlüssel von lizensierten Playern zuerst den CSS Schlüssel dekodieren und dann die Disc lesen. Im XingDVD Software Player (RealNetworks) war der der Schlüssel für CSS nicht verschl”usselt (nur die Software war “verwischt” - das ist aber ein grundsätzliches Problem !!!) und konnte einfach herausgelesen werden.

Bei Softwarelösungen muss der Schlüssel allerdings immer irgendwo im Computer in unverschlüsselter Form liegen und kann mit entsprechendem Wissen herausgelesen werden. Diese Entwicklung hat die Einführung/Verbreitung von schreibbaren DVD wesentlich verzögert.

BLU RAY: Sicherheitskonzept I

- Region Code
- High-bandwidth Digital Content Protection (HDCP)
- Advanced Access Content System (AACS)
- Self-Protecting Digital Content / BD+
- BD-ROM Mark

BLU RAY: Region Code



Werden nur von der Player Software geprüft, Publisher können selbst entscheiden ob das eingeschalten wird (70% der Titel sind Region Code frei, variiert je nach Studio).

BLU RAY: HDCP

Eine BluRay Disk kann ihre Wiedergabe in geringerer Auflösung erzwingen bzw. sie völlig unterbinden, wenn kein korrekter HDCP-link zum Anzeigegerät aufgebaut wurde. Bevor Daten übertragen werden, überprüft das Übertragungsgerät ob das Empfängergerät zum Empfang autorisiert ist. Bei autorisierter Übertragung werden die Daten verschlüsselt übertragen. Key-revocation bei kompromittierten Geräten ist ebenfalls möglich.

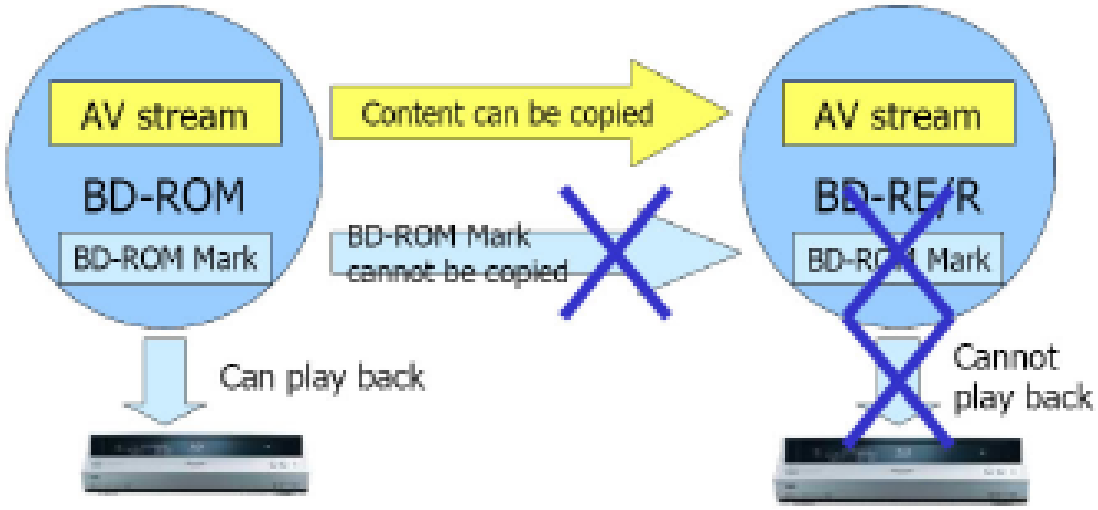
Jedes HDCP Gerät hat eine eindeutige Menge an 40 56-bit Schlüsseln. Zusätzlich gibt es den KSV (key selection vector) mit 40 bits (für jeden Key ein bit). Während der Authentifizierung zwischen zwei Geräten werden die KSV ausgetauscht und in Abhängigkeit vom KSV des anderen Geräts werden die Schlüsser binär addiert. Die Konstruktion der Schlüssel erzeugt in beiden Geräten den identischen 56-bit Key, der in einem Stream Cipher verwendet wird die Daten bei der Übertragung zu verschlüsseln.

Ist ein bestimmtes Gerät kompromittiert, wird das ensprechende KSV auf die revocation Liste von neuen Disks gebrannt (DSA signiert um revocation von legitimen Benutzern zu verhindern). Während der Authentifizierung wird diese Liste geprüft.

2001 grundsätzlich gebrochen, 2010 wurde ein master key publiziert, der das HDCP revocation System neutralisiert.

BLU RAY: BD-ROM Mark

Es handelt sich um eine kleine Menge kryptographischer Daten, die separat von den anderen Daten gespeichert ist. 1:1 Kopien einer Disk können nicht abgespielt werden. Für das Einbetten der Daten wird ein lizensiertes Gerät benötigt. Die gespeicherten Daten beinhalten die Volume ID die für die AACS Entschlüsselung benötigt wird.



Die Technik basiert auf Ungenauigkeiten bei der Drehzahl / Geschwindigkeit beim Kopieren. Die Position der Daten bleibt bei gepressten Disks gleich, bei neu gebrannten wird sie verändert. Die Beschreibung der erwarteten Position der Markierung ist auf der Disk und kann durch deren digitale Signierung nicht angegriffen werden.

BLU RAY: Self Protecting Digital Content / BD+

DB+ ist eine virtuelle Maschine auf Playern, die es Content Providern ermöglicht ausführbaren Code auf den Disks zu speichern. Ablauf:

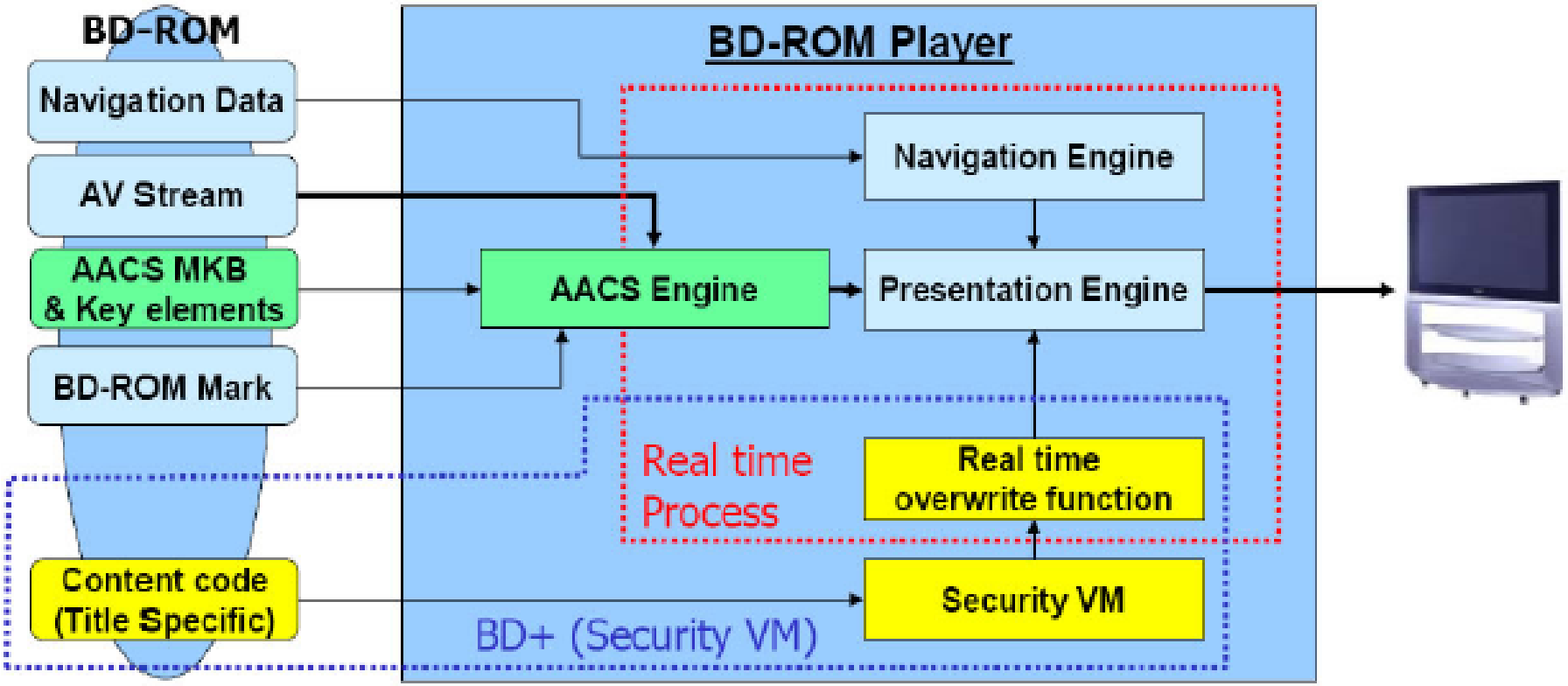
1. VM startet beim Einlegen der Disk

2. Mögliche Funktionen:
   - Überprüfen des Players - memory footprints werden verwendet um Player Manipulationen zu erkennen.
   - Überprüfung der Intergrität der Schlüssel des Players.
   - Ausführung von Code um ein als unsicher erkanntes System zu patchen (kann von Herstellern solcher Systeme bereitgestellt werden).
   - Transformation von Audio und Video Daten. Teile der Daten können nur von BD+ Dekodiert werden.
   - Korrektur / Überspringen von fehlerhaften Segmenten
   - Einfügen von Watermarks (siehe später wofür)

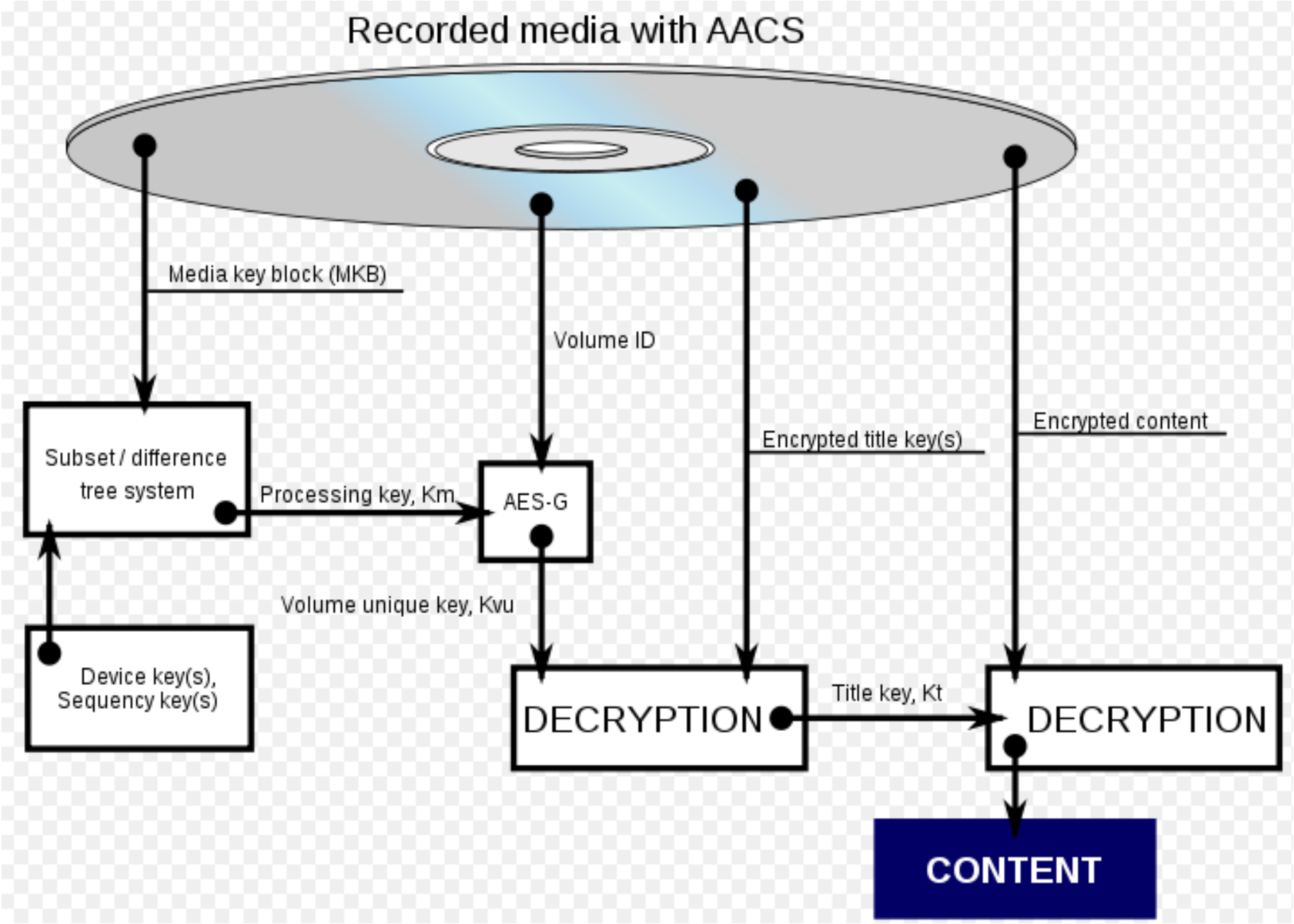
3. Beim Entfernen der Disk wird die VM beendet und die Code-Teile werden aus dem Player gelöscht der dadurch wieder in den Ausgangszustand versetzt wird.

2008: Slysoft AnyDVD HD 6.4.0.0

BLU RAY: Gesamtsystem



BLU RAY: AACS Übersicht



BLU RAY: AACS

AACS verwendet 128-bit AES Verschlüsselung um den Inhalt der Disk (Video, Audio) zu schützen. Zur Entschlüsselung werden meist mehrere “title keys” verwendet, die ihrerseit verschlüsselt sind. Diese Title Keys werden entschlüsselt durch eine Kombination des “media key” (kodiert im sogenannten “media key block” MKB) und der volume ID der Disk (eine physikalische Seriennummer die durch die DB-ROM Mark eingebettet wird. Das Ergebnis, der “volume unique key”, entschlüsselt die title keys.

Im MKB sind unterschiedlich verschlüsselte media keys durch ein *broadcast encryption* Verfahren namens “subset difference set” abgespeichert das insebsondere ermöglicht, einzelne Player zu sperren (später mehr dazu). Durch eine Veränderung der MKB können einzelne oder mehrere Player eine Disk nicht mehr abspielen.

Sequence Key Blocks (SKB): bis zu 32 kurze Videosegmente können mit zusätzlichen keys verschlüsselt sein – an jedem dieser 32 Segmente gibt es je 8 unterschiedliche Daten von denen nur eine Variante von den sequence keys entschlüsselt werden kann. Verschiedene Player dekodieren daher unterschiedliche Versionen der Daten, was einen Fingerprint eines unsicheren Players liefert, der mit der nächsten MKB Variante gesperrt werden kann. Identifikation der Versionen durch watermarks.

Managed Copy: wie bei DVD verschiedene Varianten.

BLU RAY: Subset Differerence Set I

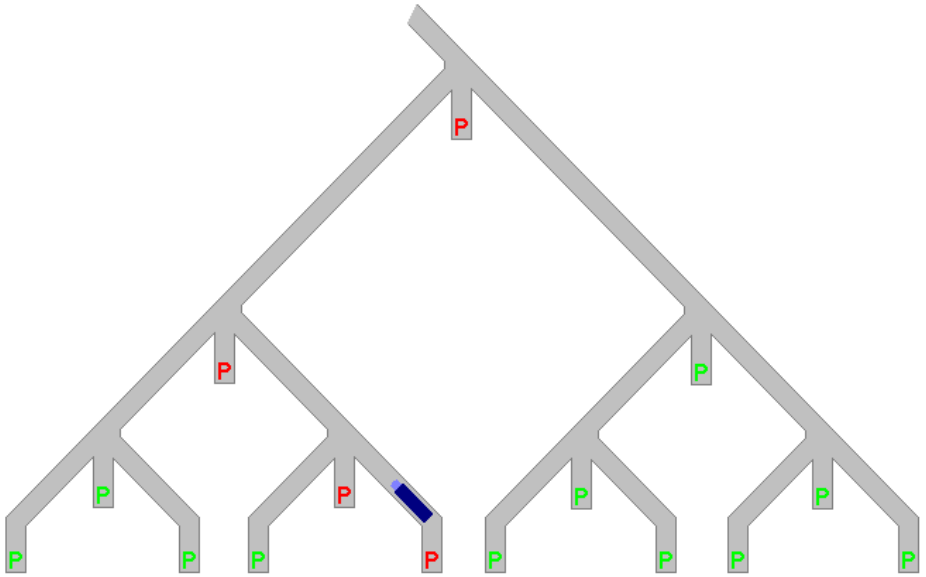
MKB ist auf der Disk gespeichert, im Player gibt es eine kleine Menge von “device keys”, die benutzt werden um im MKB den benötigen “processing key” zu berechnen, der auf einen “C-Wert” (ein verschlüsselter media key) aus dem MKB angewendet wird um den media key zu generieren.

MKB ist im wesentlichen ein sog. “subset-difference-tree” System: eine baumartig organisierte Sammlung von verschlüsselten processing keys und device keys. Die device keys dienen dazu, processing keys herzuleiten und da es nur wenige solcher device keys im Player gibt können von einem bestimmten Player nur wenige processing keys erreicht bwz. erzeugt werden. Dies wird verwendet um durch Veränderung des MKB einzelne Player “auszuschalten”.

Im Folgenden wird ein einfaches Modell von fahrenden LKWs in einem Baum verwendet. Fahren bedeutet hier die Anwendung von one-way functions (AES-G3) auf den device key oder sub-Device key was entweder zum processing key oder weiteren sub-Device keys führt.

BLU RAY: Subset Differerence Tree System I

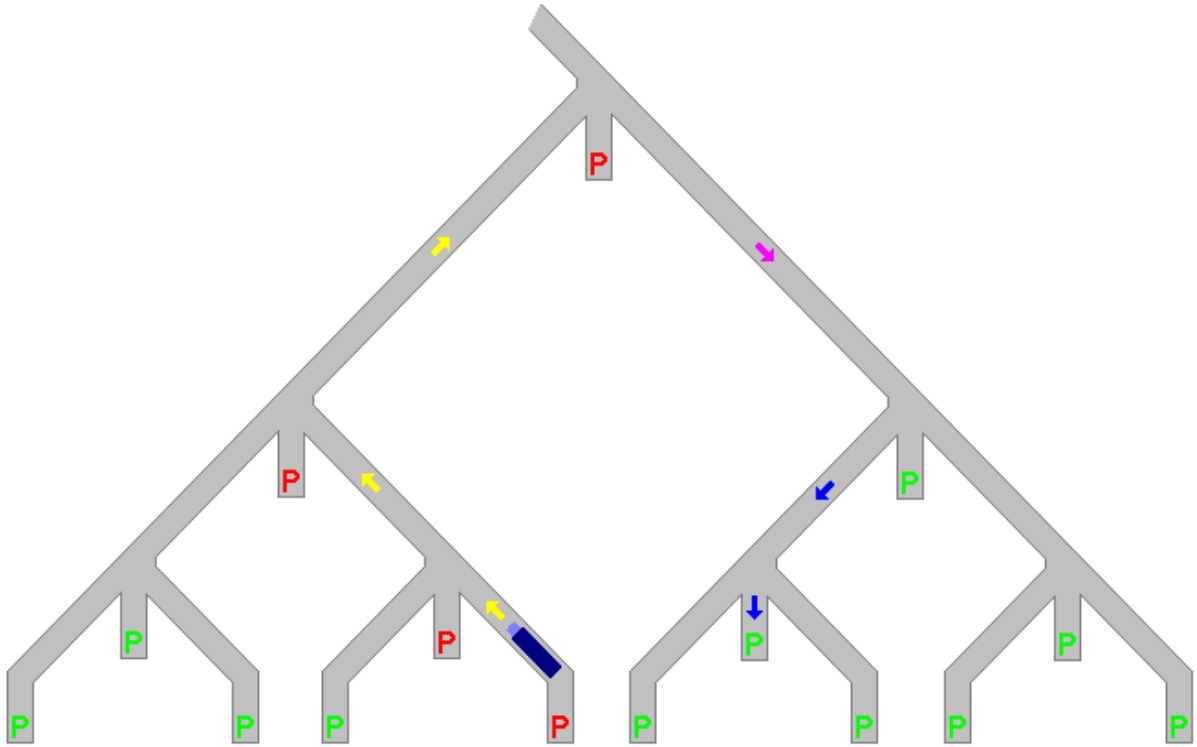
Ein LKW kann maximal eine 90 Grad Kurve machen und er kann nicht umkehren. Die grünen Parkplätze sind erreichbar und die roten nicht (vom Startpunkt). Gegeben ist ein LKW (der Softwareplayer) mit einigen Paketen und Anweisungen. Parkplätze sind processing keys im MKB die sich nicht verändern, nur das Ziel kann sich verändern (welcher Parkplatz muss angesteuert werden).



Die Anweisungen (welcher Parkplatz) stehen im MKB file. Dort auf dem Parkplatz gibt es die Information (der processing key) wie die Schachtel mit dem C im LKW geöffnet werden kann. Darin ist der media key.

BLU RAY: Subset Differerence Tree System II

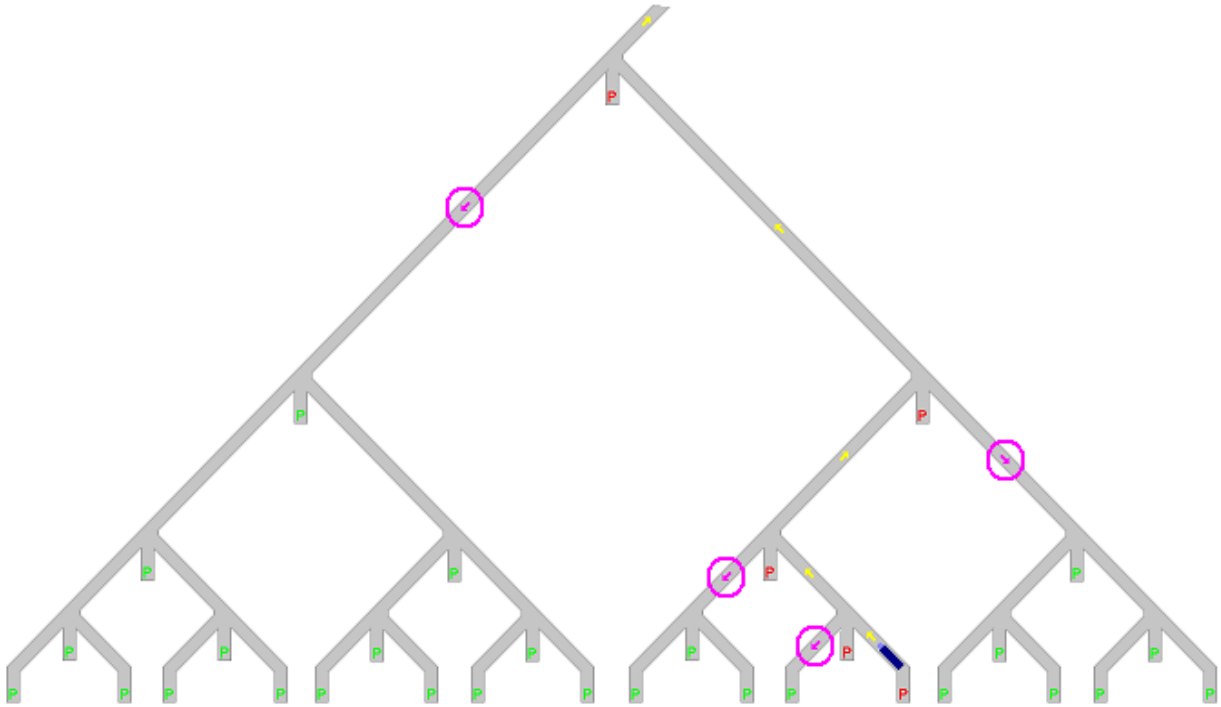
Man kann nicht jeden Parkplatz erreichen. Hat man nun eine Startposition von der aus der Parkplatz nicht erreichbar ist, kann der media key nicht erzeugt werden. Es gibt sehr viele LKWs mit Paketen darin – fällt ein LKW in Ungnade, wird ein Parkplatz gewählt, zu dem dieser LKW keinen Zugang hat. Er ist “revoked”.



Im Baum fährt der LKW zuerst nach Norden und erst dann nach Süden. Tatsächlich wird in AACS nur nach Süden gefahren, es wird an den Punkt gesprungen, von wo weg man nach Süden fährt. Diese Punkte im Baum sind die device keys.

BLU RAY: Subset Differerence Tree System III

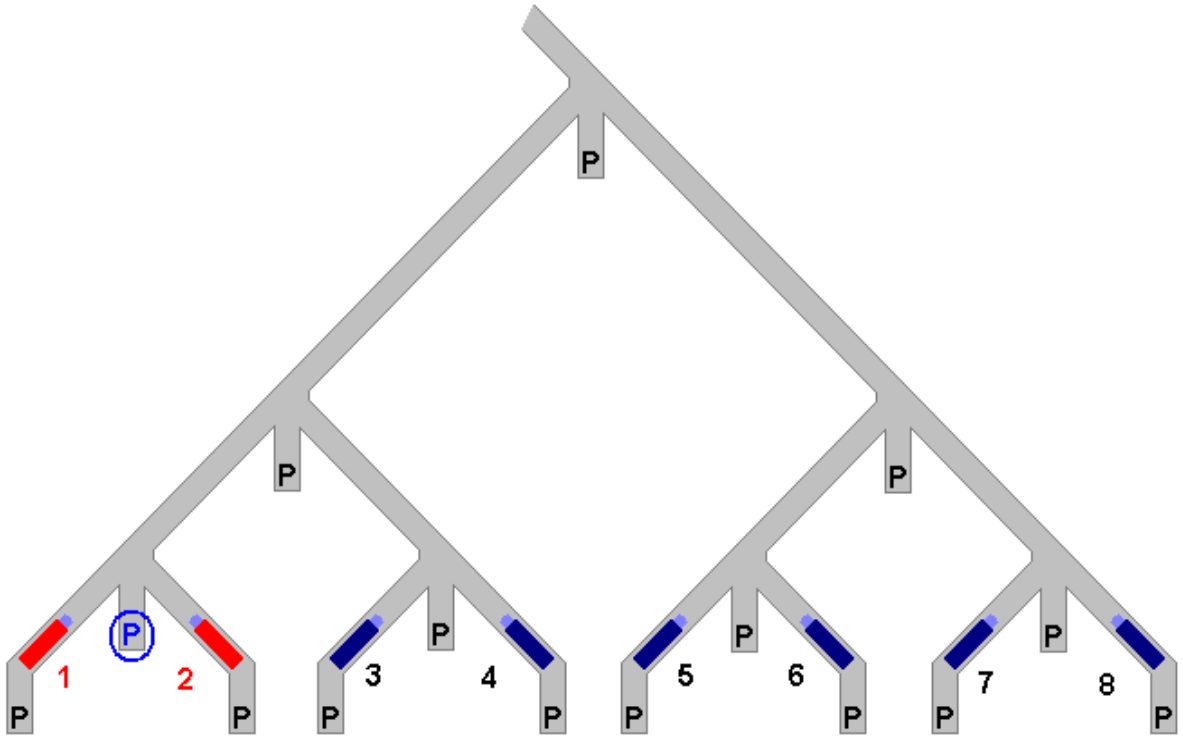
Diese Darstellung ist herausgezoomt – es ist klar ersichtlich, dass es mehrere device keys gibt von denen aus der Weg nach Süden möglich ist, allerdings ist nur einer der korrekte Einstiegspunkt um zum processing key zu gelangen.



Um herauszufinden welcher device key benötigt wird, muss man die Informationen (den MKB) konsultieren. Dann kann man an die entsprechende Stelle im Baum “springen” und den processing key generieren.

BLU RAY: Subset Differerence Tree System IV

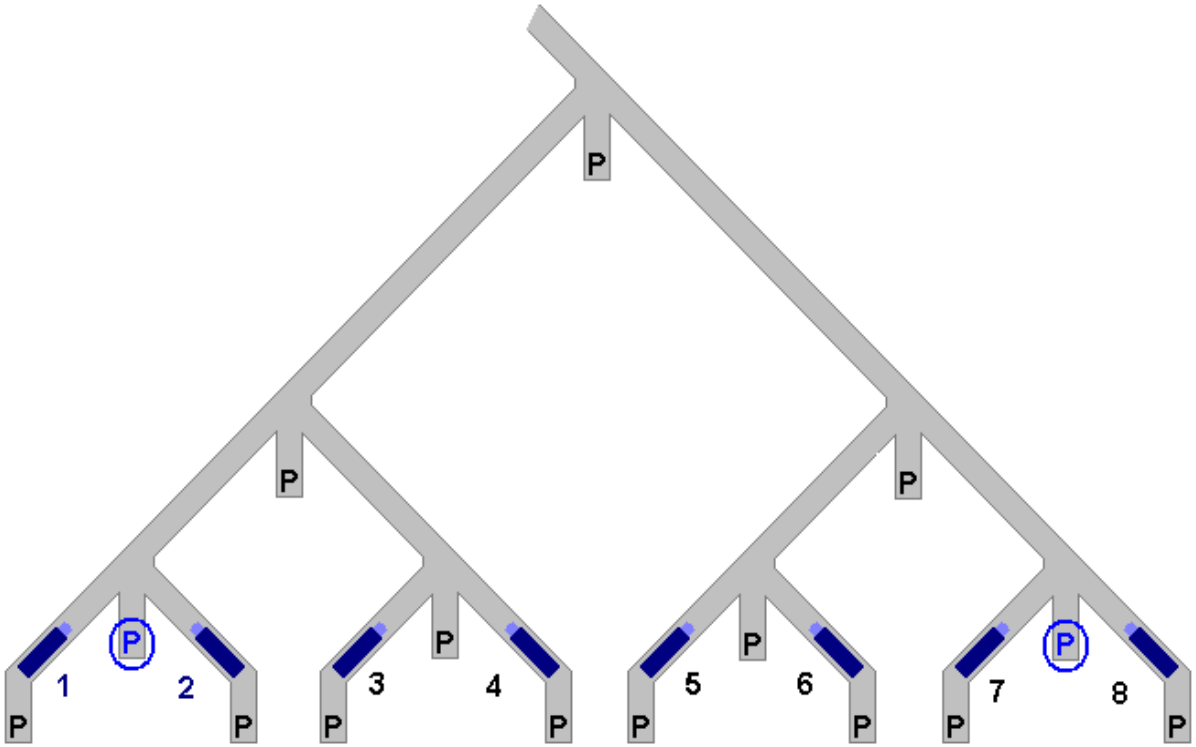
Um einzelne Player zu sperren (“revocation”) werden processing keys gewählt, die von diesen Playern (=LKWs) nicht erreicht werden können.



Im Beispiel ist der blau eingekreiste processing key das Ziel, LKW 1 & 2 werden gesperrt. Es bleibt die Frage wie LKWs gesperrt werden, die nicht benachbart sind oder wie mehrere gesperrt werden ?

BLU RAY: Subset Differerence Tree System V

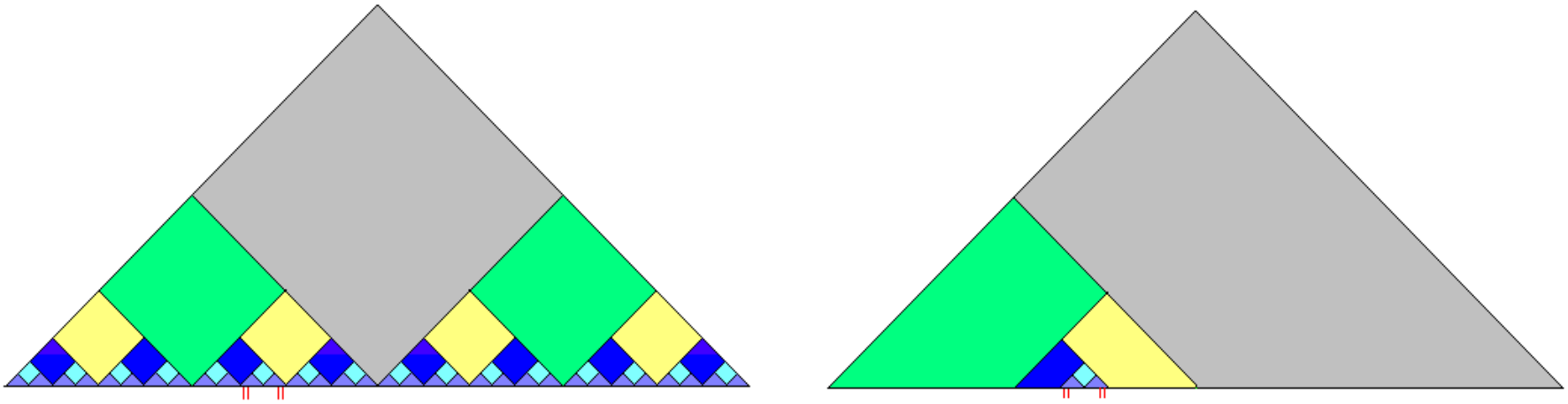
Die Idee wäre für die im Beispiel gezeigten LKWs 1,2,7,8 eine Sperrung durchzuführen, indem zwei processing keys gewählt werden, die nicht erreichbar sind.



Das funktioniert nicht, weil die LKWs zum jeweils weit entfernten processing key gelangen können, es ist also tatsächlich kein Player gesperrt worden. Das motiviert warum es nicht genug ist, einen grossen Baum zu haben.

BLU RAY: Subset Differerence Tree System VI

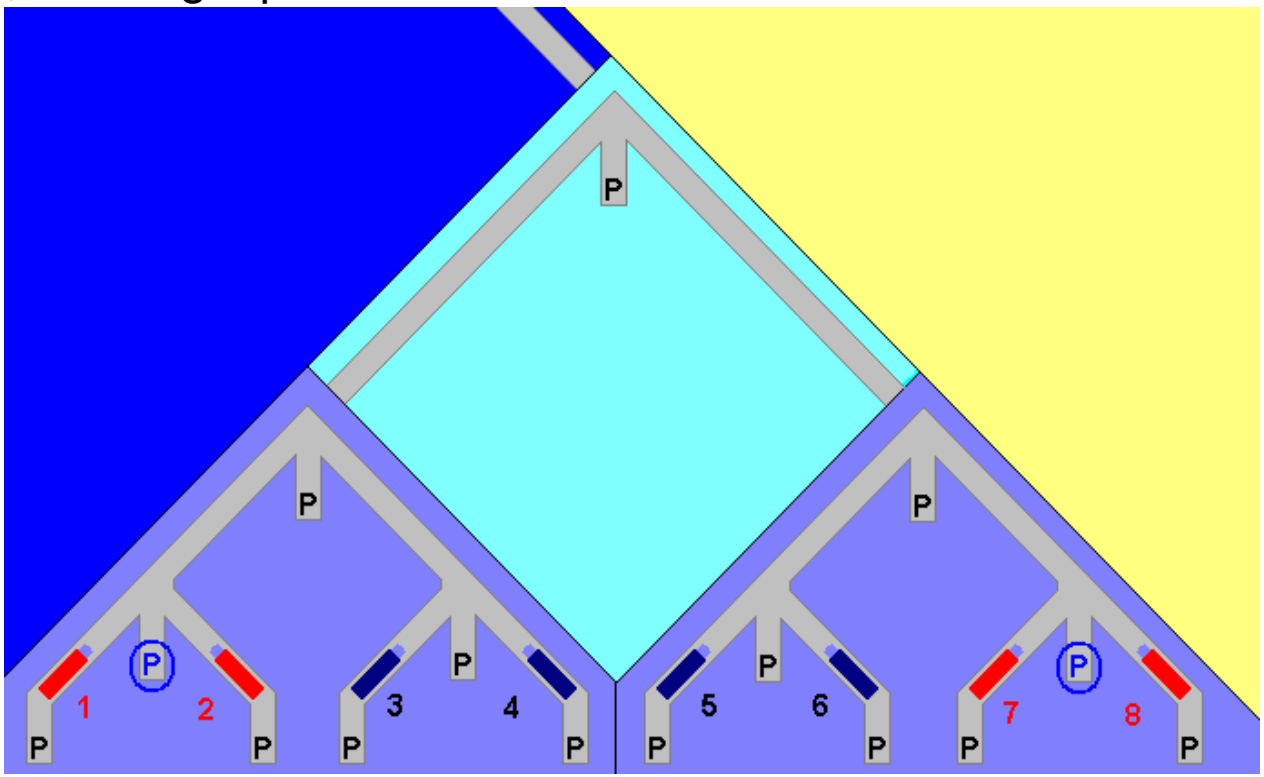
Die Lösung ist verschiedene Baum-Ebenen zu haben (tatsächlich 22), die halbe Grösse für jede Stufe nach oben aufweisen.



In der Graphik ist jede Farbe ein Stockwerk, die kleineren Stockwerke sind über den Grösseren. Die roten Punkte sind die Positionen der LKWs die gesperrt werden sollen. Die LKWs können mit einem “Aufzug” in ihr Stockwerk gelangen. Die Darstellung rechts zeigt die Teile des MKB die für die betrachteten LKWs relevant sind.

BLU RAY: Subset Differerence Tree System VII

In der Darstellung ist ersichtlich dass es keine direkte Verbindung mehr zwischen den beiden Teilbäumen gibt, d.h. hier wurden nun tatsächlich LKWs 1,2,7,8 gesperrt wogegen 3,4,5,6 nicht gesperrt worden sind.



Die unterschiedlichen processing keys (also Parkplätze) sind in der Tat verschieden, sodass auch der C-Wert der verschiedenen Teilbäumen entspricht unterschiedlich sein muss um den gleich media key zu generieren.

Partielle Verschlüsselung: Bilder

Bei der Bildverschlüsselung geht es um klassische Standbilder (still images) oder I-Frames in Videostreams.

Grundsätzlich ist zu unterscheiden ob die Verfahren im Bildbereich ansetzen, oder ob davon ausgegangen werden muß, daß die Daten in Form eines Bitstreams vorliegen (wobei durch Kompression oder Dekompression jeweils ein Fall aus dem Anderen erzeugt werden kann; das kann aber unerwünscht sein wegen Rechenaufwand oder Informationsverlust).

Selektive oder Partielle Verschlüsselung hat zwei mögliche Ziele:

- Reduktion des Rechenaufwands durch teilweise Verschlüsselung (Tradeoff Sicherheit - Komplexität)
- Erhalten der Bitstreamstruktur durch nicht-verschlüsseln von Headerdaten (format compliance für Transcoding und Error Resilience, Resynchronisierung)

Partielle Bildverschlüsselung: Anwendungsszenarien für Komplexitätsreduktion

Rechenzeitreduktion scheint in jedem Fall eine gute Idee zu sein (Beispiel: VoD Server mit individueller Verschlüsselung für jeden Client bei vielen Clients). Die Sicherheit nimmt jedoch natürlicherweise auch ab. Dieser Nachteil sollte nur bei *signifikanter* Rechenzeitreduktion in Kauf genommen werden. Vermutlich eignen sich nicht alle Anwendungsszenarien für selektive Verschlüsselung.

Klassifikationskriterien für Anwendungsszenarien:

- Lossless oder Lossy Datenformat
- Daten sind als Bildaten oder als Bitstream gegeben

Partielle Bildverschlüsselung: Lossless oder Lossy

Lossy Formate erreichen eine wesentlich höhere Kompressionsrate und sind daher wenn möglich vorzuziehen. Allerdings meist höherer Rechenaufwand !

Gründe warum man doch auf verlustfreie Datenrepräsentierung zurückgreift:

- Die Applikation erlaubt keinen Datenverlust (z.B. Gesetzeslage und Diagnosegenauigkeit bei medizinische Bildgebung, GIS Daten und sonstige Pläne)
- Durch die geringe Rechenleistung der Hardware ist überhaupt keine Kompression möglich und man bleibt im Bildbereich
- Durch die am Übertragungskanal zur Verfügung stehende hohe Bandbreite ist keine verlustbehaftete Kompression notwendig

Partielle Bildverschlüsselung: Bildbereich oder Bitstream

Egal ob verlustfrei oder verlustbehaftet komprimiert wird: Kompression muß IMMER vor einer Verschlüsselung durchgeführt werden. Statistische Eigenschaften von Verschlüsselten Daten erlauben keine Kompression mehr und die Datenreduktion der Kompression reduziert den Aufwand des Verschlüsselunbsschrittes.

1. On-line Applikationen: Die Verarbeitungskette bezüglich Verschlüsselung beginnt mit den Bilddaten, meist nach der Aufnahme der Daten. Beispiele: Videoconferencing, Digitale Kamera, Überwachungsapplikationen.

2. Off-line Applikationen: Die Verarbeitungskette bezüglich Verschlüsselung beginnt mit dem Bitstream. Sobald Bilddaten einmal gespeichert oder übertragen wurden, liegen sie meist in komprimierter Form vor. Beispiele: VoD, Bilddatenbanken, Photo CD. Diese Anwendungen sind rein retrieval-basiert.

Partielle Bildverschlüsselung: Anwendungsszenarien

	Lossy	Lossless
Bitstream	Szenario A	Szenario B
Image	Szenario C	Szenario D

Notation: Im Folgenden bezeichnet t die Verarbeitungszeit, E ist die Verschlüsselung, SE die partielle Verschlüsselung, C die Kompression, P das Preprocessing für SE zur Extraktion/Identifikation der zu verschlüsselnden Teile, $>>$ heißt wesentlich größer. Achtung: t ist nicht equivalent zu Komplexität ! Wird Kompression in Hardware ausgeführt und Verschlüsselung ist Software, ist t für die Kompression wesentlich kleiner als für die Verschlüsselung, das Umgekehrte tritt meist auf wenn beides in Software ausgeführt wird (siehe Szenario C).

Partielle Bildverschlüsselung: Szenario A & B

Gegeben sei ein Bitstream B (durch vorangegangene Kompressison). Um den Einsatz selektiver Verschlüsselung zu rechtfertigen muß folgende Bedingung erfüllt sein:

$$t(E(B)) >> t(P) + t(SE(B)) \tag{1}$$

P ist hier die Identifikation relevanter Teile des Bitstreams. $t(P)$ kann daher sehr unterschiedlich sein: bei einem embedded Bitstream oder einem Bitstream mit mehreren quality layers ist dieser Aufwand nahezu nicht vorhanden (der erste Teil bzw. der base layer wird verschlüsselt, siehe auch “transparent encryption”), im ungünstigeren Fall muß teilweise dekodiert oder zumindest der Bitstream geparsed werden um die signifikantne features zu identifizieren. Im günstigen Fall ist $t(P)$ jedenfalls vernachlässigbar, $t(E(B)) >> t(SE(B))$ ist erfüllbar und die partielle Verschlüsselung wäre profitabel. Im ungünstigen Fall wäre $t(P)$ jedenfalls kritisch zu bewerten !

Partielle Bildverschlüsselung: Szenario C

Gegeben seien die (Roh)Bilddaten I (z.B. durch vorangegangene Aufnahme) die verlustbehaftet komprimiert werden. Um den Einsatz selektiver Verschlüsselung zu rechtfertigen muß folgende Bedingung erfüllt sein:

$$t(C(I)) + t(E(C(I))) >> t(C(I)) + t(P) + t(SE(C(I))) \quad (2)$$

P ist identisch zu vorher und die diesbezüglichen Überlegungen gelten analog. Auch wenn $t(P) = 0$, ist die Ungleichung (2) kaum zu erfüllen da für fast alle Symmetrischen Verschlüsselungsverfahren und verlustbehafteten Kompressionsverfahren $t(C(I)) >> t(E(C(I)))$ gilt (wenn beide in Software oder Hardware ausgeführt werden). Daher spielt der Unterschied zwischen $t(E(C(I)))$ und $t(SE(C(I)))$ meist keine Rolle. Für hohe Kompressionsraten ist dieser Effekt stärker ausgeprägt (da der zu verschlüsselnde Bitstream kleiner ist). In diesem Fall rechtfertigt der marginale Zeitgewinn die geringere Sicherheit nicht !

Partielle Bildverschlüsselung: Szenario D

Gegeben seien die (Roh)Bilddaten I (z.B. durch vorangegangene Aufnahme) die nicht notwendigerweise komprimiert werden müssen. Gilt $t(C(I))+t(E(C(I)))<t(E(I))$ oder ist Kompression durch andere Gründe notwendig (z.B. beschränkte Kanalkapazität), werden die Daten lossless komprimiert und die Bedingungen von Szenario C “tritt in Kraft”. Da $t(C(I))>>t(E(I))$ für die meisten symmetrischen Verschlüsselungsverfahren und lossless Kompressionsverfahren gilt, wird Kompression zur Komplexitätsreduktion kaum ausgeführt. Außerdem ist die Datenreduktion im lossless Fall wesentlich geringer was den Beitrag von $t(E(C(I)))$ wichtiger machen würde.

Um den Einsatz selektiver Verschlüsselung (ohne Kompression) zu rechtfertigen muß folgende Bedingung erfüllt sein:

$$t(E(I))>>t(P)+t(SE(I)) \tag{3}$$

P ist die Identifikation/Extraktion relevanter Bildteile, was auf verschiedene Arten geschehen kann (siehe [32], z.B. Wavelets, DCT, Bitplanes, ...). Wichtig für die Erfüllung der Ungleichung (3) ist daß $t(P)$ klein bleibt !

Beispiel: Verschlüsselung eines J2K Bildes mit AES

Das folgende Beispiel (aus Pommer und Uhl [23]) dient als Illustration der Problematik partieller Verschlüsselung im Szenario C. Gegeben sei das Bild in Rohdatenform. Das Bild wird zuerst J2K komprimiert und anschließend AES verschlüsselt. Verwendete Software:

- J2K: `http://jj2000.epfl.ch`
- AES: `http://www.esat.kuleuven.ac.be/~rijmen/rijndael/`

Als Testbeispiel wählen wir ein $512 \cdot 512$ Pixel 8 bpp Grauwertbild und setzen eine typische Kompressionsrate von ca. 26 (80000 Bit). `[]` bezeichnet eine Array lookup Operation, = ist eine Zuweisung, und `^ & + %` sind bitweises exklusives oder, bitweises und, Addition und Modulo Operation.

Partielle Bildverschlüsselung: AES Komplexität

name	[ ]	=	^ & + %
KeyAddition	32	16	16
ShiftRow	80	32	32
Substitution	48	16	
MixColumn	136	32	144
128 bit key, 1 block	2858	944	1792
192 bit key, 1 block	3450	1136	2176
256 bit key, 1 block	4042	1328	2304
256 bit key, 80000 bit	2526250	830000	1440000

Für die Verschlüsselung der 80000 Bit werden daher ca. 4796250 Operationen benötigt.

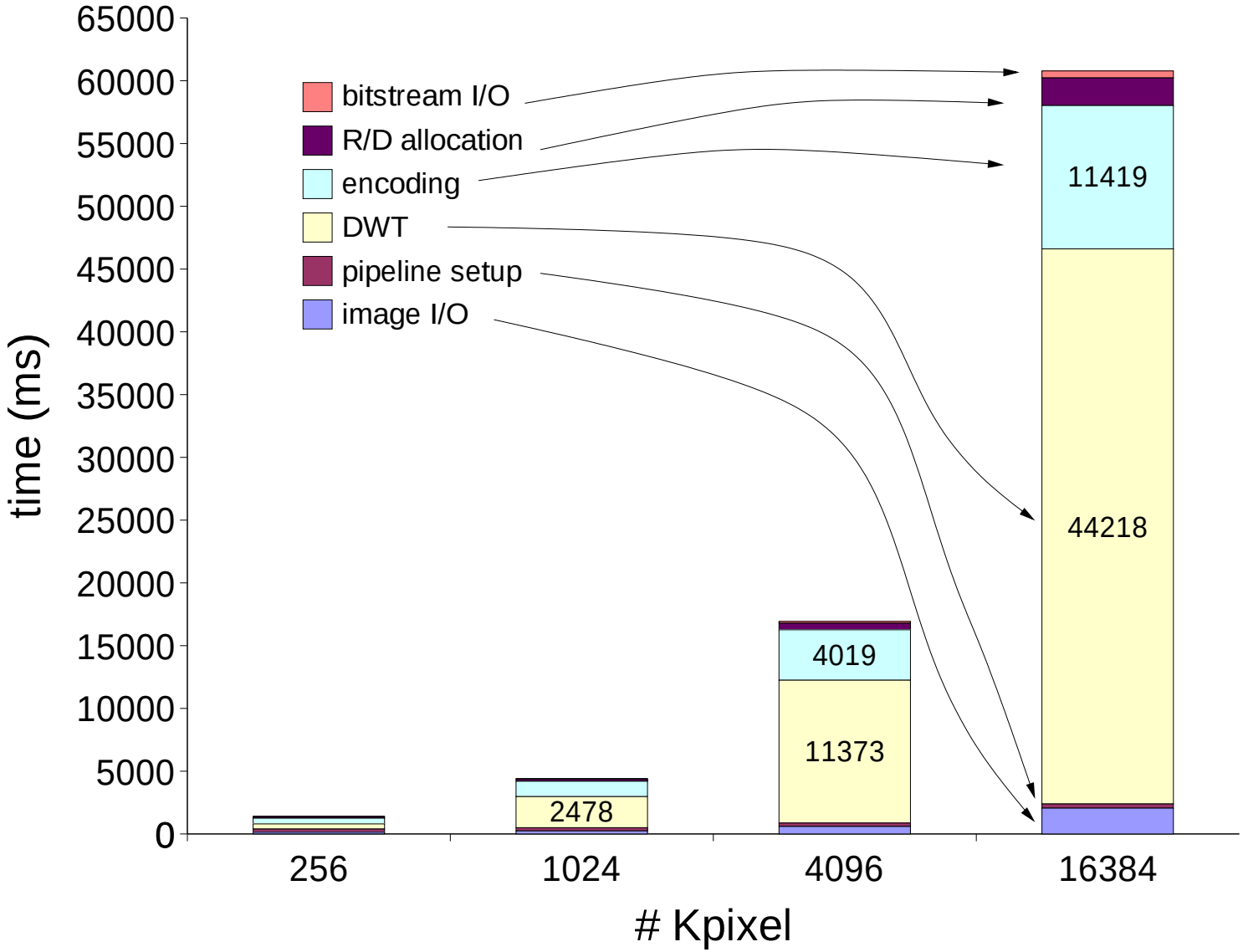
Partielle Bildverschlüsselung: Wavelet Transformation Komplexität

image size (length of one side) filter size	N n	level 1 decomposition, 1 line level 1 decomposition, total	$\frac{N}{2} * n$ $2 * \frac{N}{2} * \frac{N}{2} * n = \frac{N^2 n}{2}$
1 line	$N * n$	level i decomposition, 1 line	$\frac{N}{2^i} * n$
1 image (=total)	$2 * N * N * n = 2N^2 n$ (step size = 2, high+lowpass, horizontal+vertical filtering)	level i decomposition, total	$2 * \frac{N}{2^i} * \frac{N}{2^i} * n = \frac{2N^2 n}{2^{2i}}$

Partielle Bildverschlüsselung: Wavelet Transformation Komplexität

example values	$N = 512$ pixel side length, $n = 8$
1 operation includes	1 addition $+$, 1 multiplication $*$, 2 array lookups $[\,]$
1 decomposition	$2N^2n = 4\,194\,304$ operations
standard wavelet decompo-sition	$\sum_{i=1}^{5} \frac{2N^2n}{2^{2(i-1)}} = 5\,586\,944$ operations

Partielle Bildverschlüsselung: J2K Laufzeit Verhalten



Partielle Bildverschlüsselung: J2K Overall

Betrachtet man den Fall der J2K Defaulteinstellungen (5 Zerlegungsstufen und 7/9 biorthogonaler Filter) kommt man bei einem 512×512 Pixel Bild auf $5\,586\,944$ Operationen. Diese Anzahl muß noch mit 4 multipliziert werden da jede Operation eine Addition, eine Multiplikation und zwei Array Lookups beinhaltet.

Dazu kommen noch zumindest $\sum_{i=1}^{5} \frac{N^2}{2^{i-1}}$ Zuweisungen. Insgesamt führt das zu $22\,855\,680$ Operationen für die Filterung und ca. $31\,744\,000$ Operationen für die gesamte Verarbeitungspipeline (wenn man die 28% Verarbeitungszeit der nicht-filterungsbezogenen Operationen dazurechnet).

Zum Vergleich: bei AES hatten wir $4\,796\,250$ Operationen !

Partielle Bildverschlüsselung: J2K & AES Resume

Offensichtlich ist die Zahl der Operationen bei J2K ca. 7 Mal so groß wie bei AES. Zusätzlich ist der Speicherbedarf bei AES wesentlich geringer und das Cacheverhalten wesentlich besser (durch 128 Bit Blockgröße und einfachste Operationen wie Lookup Tables). Wavelet Transformation braucht große Speicherbereiche und hat durch die alternierende horizontal-vertikale Filterung ein extrem ungünstiges Cache Verhalten.

Testläufe mit beiden Softwareimplementierungen ergeben einen AES Anteil von unter 1% (!!!!) an der gesamten Kompressions-Verschlüsselungs Pipeline.

Daraus folgt, daß es bei einem dermaßen kleinen Anteil der Verschlüsselung an der Gesamtkomplexität keinen Sinn macht, durch partielle Verschlüsselung diesen ohnehin kleinen Anteil weiter zu verkleinern. In diesem Fall ist der Sicherheitsverlust durch partielle Verschlüsselung nicht zu rechtfertigen.

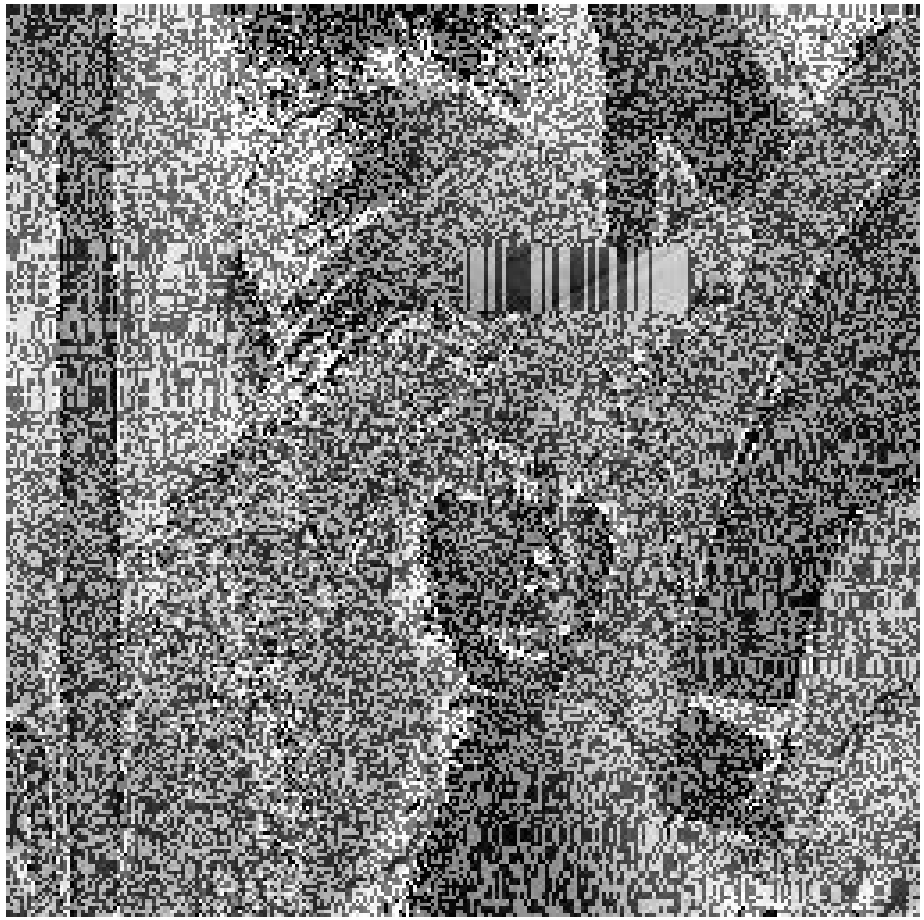
Partielle Bildverschlüsselung: Selektive Bitplane Verschlüsselung (Szenario D)

Ausgangssituation ist die Aufnahme eines digitalen Bildes unter der Vorraussetzung daß es verlustfrei verarbeitet werden soll und keine Kompression notwendig oder wünschenswert ist. Um selektive Verschüsselung profitabel und sinnvoll zu machen, muß die Identifikation/Extraktion von relevanten Features schnell sein.

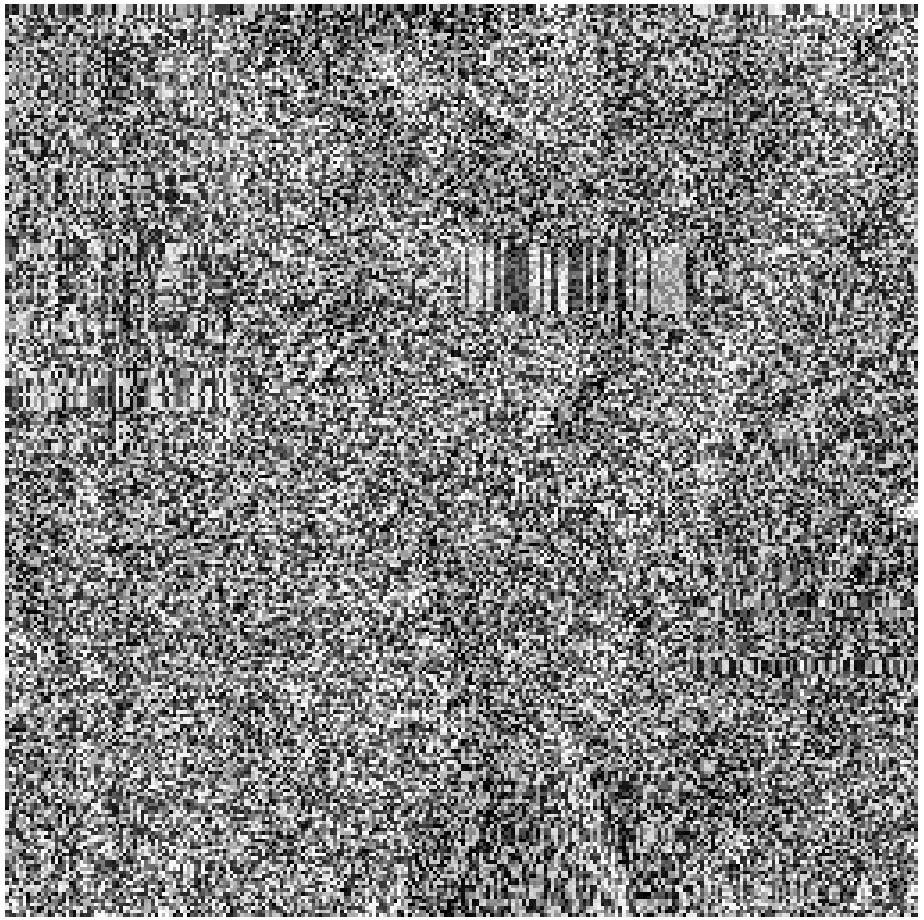
Als schnellste Methode dies zu erreichen wird in Podesser et al. [21] und Skrepth und Uhl [32] selektive Bitplane Verschlüsselung vorgeschlagen. Grundidee ist es die binäre Representation der Pixelwerte zu betrachten und eine Teilmenge der sich ergebenden Bitplanes (binäre Bilder quer durch die Bitrepräsentierung) mit AES zu verschlüsseln. Die restlichen Bitplanes werden in Plaintext übertragen/gespeichert.

Im Folgenden betrachten wir 512 $\times$ 512 Pixel Bilder mit 8 bpp (ergibt 8 Bitplanes) was zu einem minimalen selektiven Verschlüsselungsprozentsatz von 12.5 % führt (1 Bitplane). Der 128 Bit AES Block wird mit einer Viertel Bitplane Zeile ($512/4 = 128$) befüllt und abgearbeitet.

Partielle Bitplane Verschlüsselung: Beispiele 1



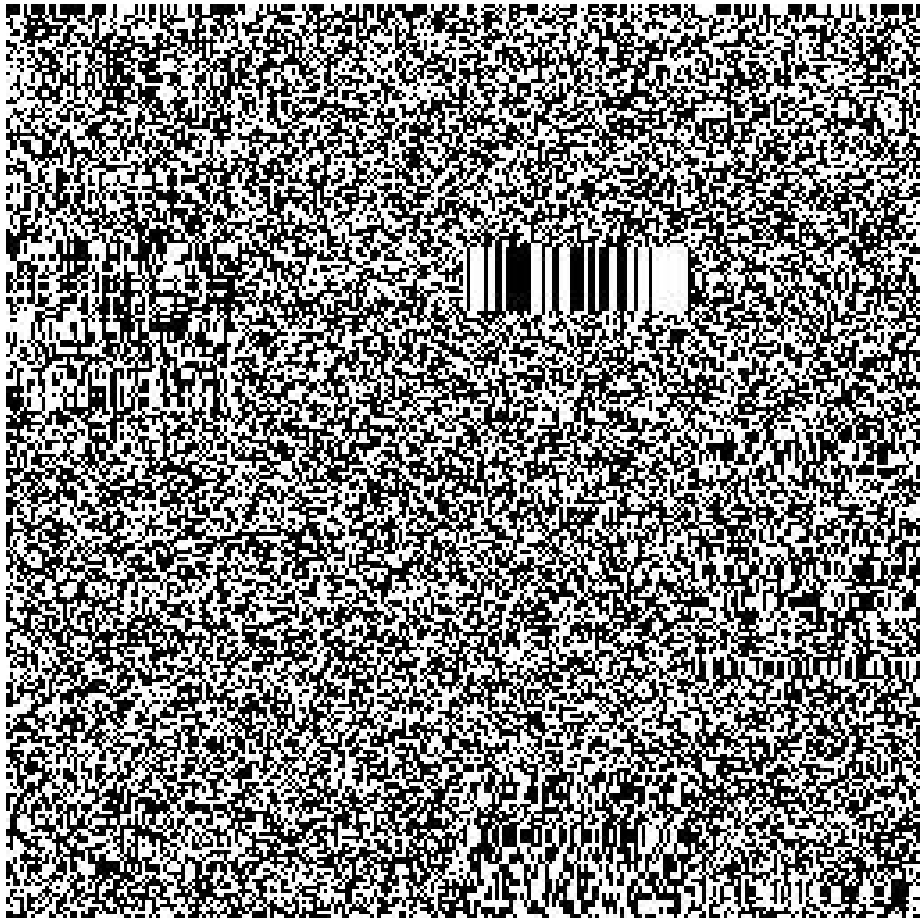
(a) 12.5% encrypted



(b) 25% encrypted, 9.0dB

Abbildung 2: Visuelle Beispiele für partielle Bitplane Verschlüsselung bei direkter Rekonstruktion.

Partielle Bitplane Verschlüsselung: Beispiele 2



(a) encrypted MSB

(b) 50% encrypted, 31.8dB

Partielle Bitplane Verschlüsselung: Eigenschaften 1

Das Barcodemuster entsteht durch identische Verschlüsselung (gleicher Key) von identischen übereinanderliegenden Viertelzeilen

Geht es um Sicherheit, muß zuallererst die MSB Bitplane verschlüsselt werden, weitere Bitplanes folgend steigender Signifikanz in der Binärdarstellung (siehe PSNR Werte in der Tabelle).

# Bitplanes	1	2	3	4	5	6	7	8
First: LSB	51	44	38	32	26	20	14	9
First: MSB	9	9	9	9	9	9	9	9

Partielle Bitplane Verschlüsselung: Eigenschaften 2

Eine eventuelle Möglichkeit die Sicherheit zu erhöhen wäre es geheimzuhalten, welche Bitplanes zusätzlich zur MSB verschlüsselt wurden. Zwei Gründe machen diese Idee kaum erfolgversprechend:

- Verschlüsselte Bitplanes nahe der LSB erhöhen die Sicherheit kaum, daher ist die Auswahl an sinnvollen Kombinationen extrem eingeschränkt.

- Die statistischen Eigenschaften von "natürlichen" und verschlüsselten Bitplanes sind sehr unterschiedlich solange die betrachteten Bitplanes nahe bei der MSB befinden (und nur da macht Verschlüsselung Sinn). Daher kann durch einfache statistische Methoden festgestellt werden, welche Bitplanes verschlüsselt wurden (die Tabelle zeigt die Anzahl der Runs von 5 identischen Bits in Tausend im Lena Bild).

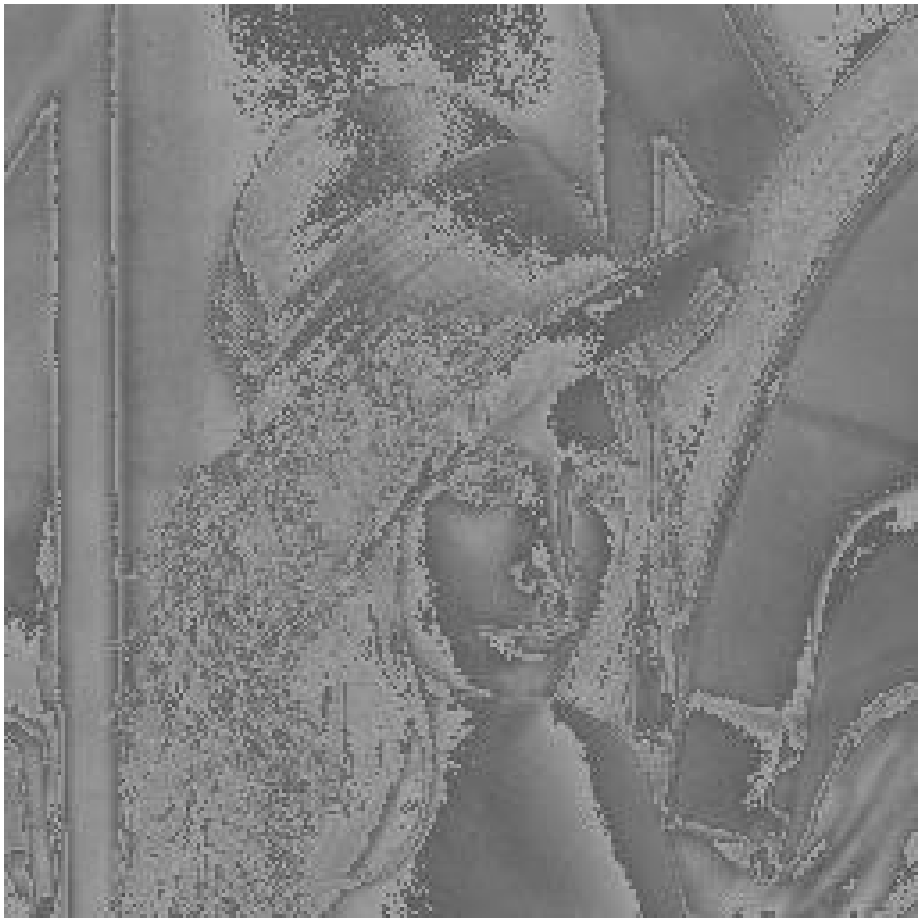
Bitplane	MSB	2	3	4	5	6	7	LSB
Plain	45	39	32	20	11	5	4	4
Encrypted	4	4	4	4	4	4	4	4

Partielle Bitplane Verschlüsselung: Ersetzungsattacke

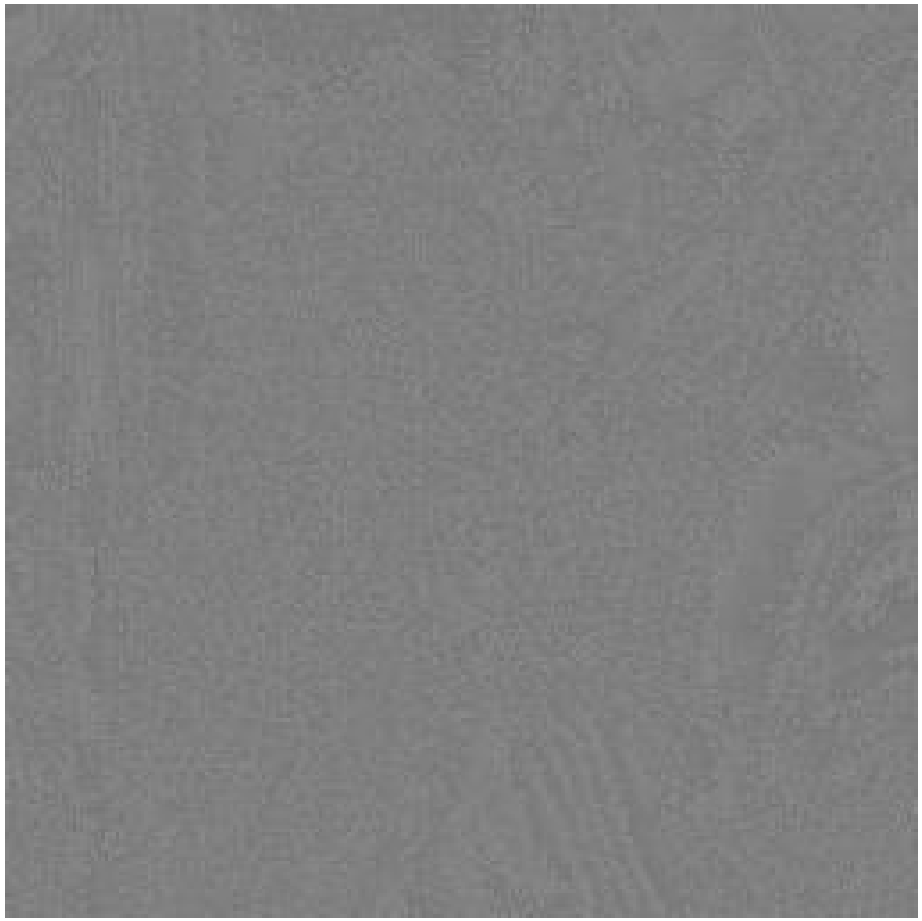
Bei direkter Rekonstruktion werden durch die verschlüsselten Teile Rauschanteile ins Bild gebracht, die die Bildqualität wesentlich beeinträchtigen. Daher werden die verschlüsselten Teile durch “typische” Daten ersetzt. Konkret wird eine konstante 0 Bitplane eingesetzt und der (vermutliche) Abfall in durchschnittlicher helligkeit durch Addition einer kosnatnte kompensiert (64 bei MSB, 96 bei zwei Planes, ...). Rekonstruktion geschieht wie gewöhnlich.

Während bei direkter Rekonstruktion die Verschlüsselung von 2 Bitplanes sicher wirkt (mit 9 dB im vorigen Beispiel), zeigen sich nach der Ersetzungsattacke wichtige Details im Bild (mit 13.2 dB, wenn auch stark verfremdet). Verschlüsselung von 4 Bitplanes (50% der originalen Datenmenge) widersteht auch der Ersetzungsattacke.

Partielle Bitplane Verschlüsselung: Ersetzungsattacke Beispiele



(c) 25% encrypted, 13.2dB



(d) 50% encrypted

Abbildung 3: Visuelle Beispiele für die Effizienz der Ersetzungsattacke.

Partielle Bitplane Verschlüsselung: Rekonstruktionsattacke

Die Grundidee ist unter Ausnutzung der unverschlüsselten Daten die verschlüsselten Bitplanes zu rekonstruieren (im betrachteten einfachsten Fall ist nur MSB verschlüsselt). Dabei wird (wieder) die überwiegende Glattheit von natürlichen Bildern ausgenutzt. In solchen glatten Regionen sind die MSB Werte benachbarter Pixel meistens identisch (außer in Regionen mittlerer Helligkeit).

Um solche Regionen zu identifizieren wird ein 2×2 Pixel Suchfenster über das Bild geschoben, wobei alle 16 möglichen MSB Konfigurationen getestet werden wobei eine Reihe von Differenzen zwischen den 4 Pixeln berechnet wird. Die kleinste Differenz wird bestimmt und die entstrechende MSB Konfiguration als Rekonstruktion gesetzt.

Kanten werden durch versuchte Kompensation im Suchfenster klar erkannt, setzen des MSB in glatten Regionen auf 0 oder 1 liefert zwei Halbbilder, die durch einfache Tests zur Rekonstruktion kombiniert werden können. Diese Attacke wird für mehrere verschlüsselte Bitplanes sehr aufwendig und ungenau.

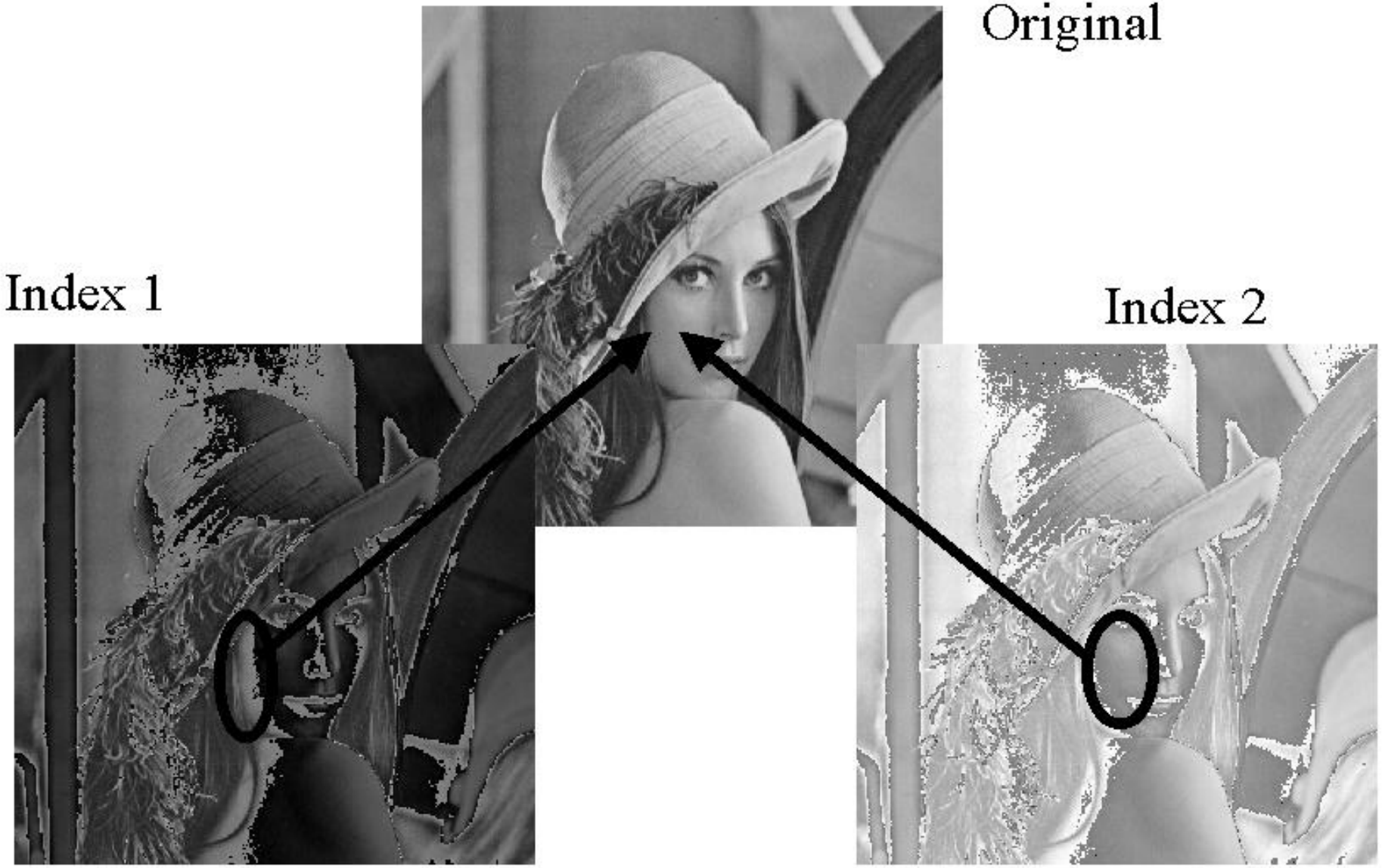
Partielle Bitplane Verschlüsselung: Rekonstruktionsattacke Beispiele 1

(a) original MSB

(b) reconstructed bitplane

Abbildung 4: MSB von Lena und rekonstruierte MSB Bitplane.

Partielle Bitplane Verschlüsselung: Rekonstruktionsattacke
Beispiele 2



Partielle Bildverschlüsselung mit Kompression

Nach folgenden Kriterien lassen sich kompressionsbasierte partielle Bildverschlüsselungsverfahren klassifizieren:

1. Art der involvierten Kompression: 8×8 Pixel DCT, Wavelettransformation, Wavelet Packets, Quadtree, ...

2. An welcher Stelle der Verarbeitungskette die Verschlüsselung durchgeführt wird:

   - Im Zuge des Kompressionsvorgangs (siehe auch Szenarios C & D):
     - ⋆ Koeffizienten werden ganz oder teilweise verschlüsselt
     - ⋆ Geheime Kompression (z.B. geheime Transformdomains)
   - Nach dem Kompressionsvorgang: der Bitstream wird verschlüsselt (siehe auch Szenarios A & B):
     - ⋆ Header Verschlüsselung
     - ⋆ Scalable/Embedded Bitstreams: ohne Bitstream Parsing
     - ⋆ Speziell: Transparent Encryption für Preview

Partielle Bildverschlüsselung im Kompressionsprozeß: Pros & Cons 1

Grundsätzlich gibt es in der Literatur unterschiedliche Meinungen wo optimalerweise die Verschlüsselung ansetzen sollte:

Liegt bereits ein Bitstream vor, müßte im Fall der Kompressions-gleichzeitigen Verschlüsselung ein zumindest teilweises Dekodieren stattfinden. CON.

Vorhandene Kompressions Hardware läßt sich nur bei bitstreambasierter Verschlüsselung einsetzen. CON.

Das Arbeiten im Transformationsbereich ist wesentlich einfacher (die relevante Information läßt sich wesentlich direkter/effizienter identifizieren). PRO.

Hier wird Verschüsselung VOR der Kompression eingesetzt. Gefahr der geringeren Kompressionsleistung droht, um das zu umgehen müssen zu verschlüsselnde Teile wohlüberlegt ausgewählt werden. CON.

Partielle Bildverschlüsselung im Kompressionsprozeß: Pros & Cons 2

Wird nur eine Permutation auf die Koeffizienten ausgeführt, können Histogramm und Energieverteilung der Koeffizienten ohne den Schlüssel zu kennen berechnet werden (z.B. für Suche in verschlüsselter Bilddatenbank). PRO.

Wird für Transcoding eine Dekodierung und Dequantisierung benötigt (für nicht skalierbare Bitstreams), kann das im Router ohne Schüsselkenntnis durchgeführt werden (zumindest prinzipiell, die eventuell nicht vorhandene Information über die Frequenzlokalisation der Koeffizienten im Fall von Permutation bleibt ein Problem). PRO.

Partielle Bildverschlüsselung im Kompressionsprozeß: DCT I

Der historisch erste Algorithmus dieser Klasse wird als *Zig-Zag permutation Algorithm* [36, 31] bezeichnet. Die Grundidee ist, die Abbildung der 8×8 Koeffizienten auf das wohldefinierte Zig-Zag Muster (fallende Frequenzanordnung) durch eine geheime Anordnung zu ersetzen.

DC-Splitting: Der DC Koeffizient würde durch seine Größe sofort identifiziert. Deshalb wird der 8 Bit Wert vor der Permutation in zwei 4 Bit Blöcke geteilt, der MSB Teil bleibt, der LSB Teil ersetzt den letzten AC Koeffizienten.

Zwei Zusatzoptionen sollen die Sicherheit erhöhen:

- DC Koeffizienten von 8 Blöcken werden vor dem DC-Splitting vereinigt und DES verschlüsselt, dann byteweise zurückgeschrieben.
- Es werden anstelle einer fixen Permutationstabelle zwei Tabellen verwendet, deren jeweiliger Einsatz durch einen Zufallsbitstream (stream cipher) für jeden Block bestimmt wird.

Partielle Bildverschlüsselung im Kompressionsprozeß: DCT I (cont.)

Der *Zig-Zag permutation Algorithm* hat folgende Eigenschaften:

- PRO: Der resultierende Bitstream ist standardkonform.
- CON: Die Größe des resultierenden Bitstreams ist im Allgemeinen wesentlich größer als die der originalen Streams (ca. 40 - 100 %). VLC und RLE sind auf Zig-Zag Scan optimiert.
- CON: Komplexitätsreduktion über schwächeres Verschlüsselungsverfahren (fixe Permutation hat Rätseleckenniveau).

Partielle Bildverschlüsselung im Kompressionsprozeß: DCT I (cont.)

2 Attacken gegen die fixe Permutation:

1. Known Plaintext Attacke: Durch einfachen Vergleich von Plaintext und den verschlüsselten Frames lassen sich die beiden Permutationslisten ermitteln. Die richtige der beiden läßt sich auf Blockbasis durch den Vergleich der Anordnung der non-zero AC Koeffizienten ermitteln (AC Koeffizienten ungleich 0 müssen sich in der linken oberen Ecke sammeln). Das Splitten und DES Verschlüsseln der DC Koeffizienten allein bietet nicht genug Sicherheit, da Bilder noch gut erkannt werden können. In [38] wird die analoge Attacke umgekehrt (bekannte Transformationskoeffizienten werden in einen Dekoder verarbeitet) als Chosen Ciphertext Attacke beschrieben.

2. Ciphertext Only Attacke: Basierend auf statistischem Wissen über die Anordnung und Größenrelationen der AC Koeffizienten kann durch Testen weniger Kombinationen der größten Koeffizienten ein brauchbares Bild erzeugt werden. Automatisierung basierend auf Glattheitsmaßen (vergleiche Nagravision und Videocrypt).

Partielle Bildverschlüsselung im Kompressionsprozeß: DCT II

Die Hauptprobleme beim *Zig-Zag permutation Algorithm* entstehen durch die Verschlüsselung von dafür in der vorgeschlagenen Art nicht geeignetem Datenmaterial (Eigenschaften die für Kompression wesentlich sind werden zerstört) und der Verwendung eines unsicheren kryptographischen Systems. Für Verbesserungen muß demnach in diesen Bereichen angesetzt werden.

- Als zu verschlüsselndes Datenmaterial werden in [30] und früheren Arbeiten von Shi und Bhargava die Vorzeichen bits (sign bits) der DCT Koeffizienten vorgeschlagen. Dies beeinträchtigt die nachfolgende Kompression nicht, da diese Folgen durch ihre hohe Entropie ohnehin kaum komprimiert werden können. Während die ersten Vorschläge auf Permutation dieser Bitfolgen beruhen (VEA algorithm, unsicher), schlägt die letzte Version (RVEA) eine DES Verschlüsselung von sign bits basierend auf einem fixen Scan durch DC und AC Koeffizienten von Luminance und Chroma Daten vor. Durch die DPCM Kodierung der DC Koeffizienten ergibt sich eine große Durchmischung der Werte. Bei diesem Verfahren werden ca. 10% der Gesamtdatenmenge verschlüsselt. Sicherheit ist durch die Verwendung von DES entsprechend, eine Ciphertext only Attacke braucht die Komplexität des Testens aller sign Bitkonstellationen, allerdings für jeden Block neu. Bitstream ist standardkonform und praktisch gleich groß.

Partielle Bildverschlüsselung im Kompressionsprozeß: DCT III

- Eine andere Vorgehensweise wird von Zeng und Lei [44] vorgeschlagen. Hier werden mehrere 8×8 Pixel Blöcke gemeinsam betrachtet. Koeffizienten von verschiedenen Blöcken die an den gleichen (Frequenz)Positionen stehen, werden gemeinsam weiterverarbeitet. Dadurch werden virtuelle Frequenzteilbänder (Subbands) erzeugt, innerhalb derer kryptographische Methoden angewendet werden. Dadurch wird eine Beeinträchtigung der Kompression vermieden. Als Verschlüsselung werden zwei (dabei allerdings eine unsichere) Methoden vorgeschlagen:

  1. Permutation durch fixe unterschiedliche Tabellen für unterschiedliche Bands (keine Resistenz gegen known Plaintext Attacken !)
  2. Sign Bit Verschlüsselung

  Während sign Bit Verschlüsselung die Bitstreamgröße nicht verändert, tritt bei Permutation eine Datenexpansion um ca. 10% auf. Sign Bit Verschlüsselung allein bietet allerdings nur beschränkte Sicherheit (einige Strukturen sind in den Bildern erkennbar).

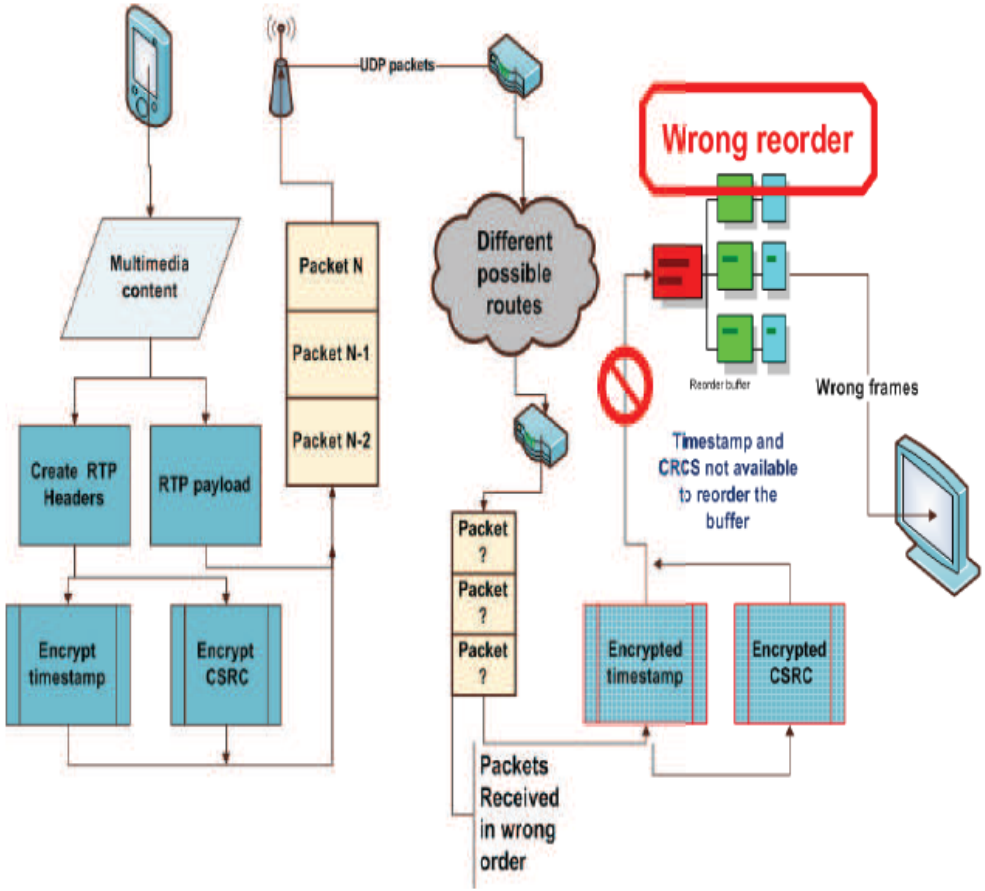
Bitstreambasierte Partielle Verschlüsselung: Headerverschlüsselung

Ein naheliegender Ansatz ist es, bei der Verschlüsselung eines Bitstreams nur die Headerdaten zu schützen um das Bildformat zu verschleiern oder das Dekomprimieren zu verhindern.

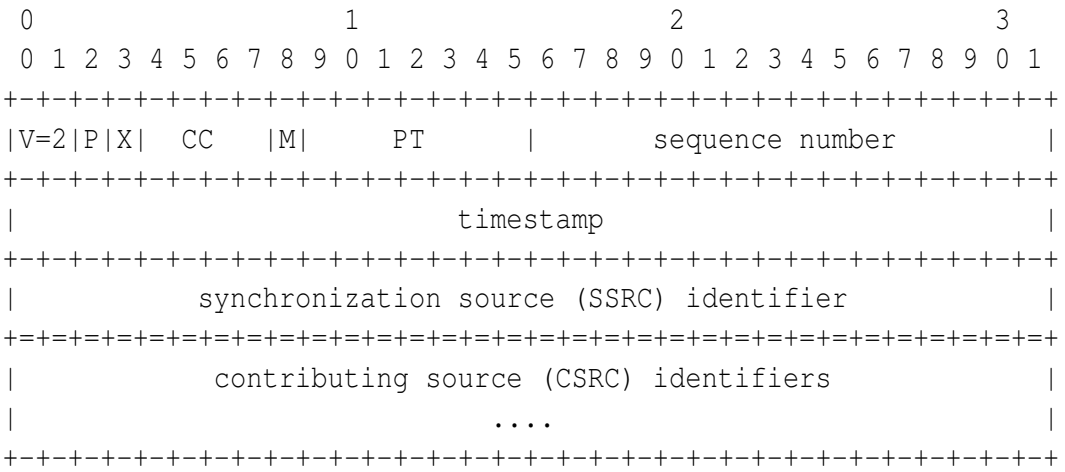
Der Aufwand für die Verschlüsselung kann hier sehr klein gehalten werden. Eine alleinige Verschleierung des Bildformates ist allerdings nicht ausreichend, da aufgrund von statistischen Eigenschaften leicht die Art des zugrundeliegenden Verfahrens erkannt wird (außer die Nichtanwendbarkeit von Standarddekompressionstools ist alleiniges Ziel). Werden tatsächlich Headerinformationen geschützt, ist es für die Sicherheit wesentlich, daß nicht nur Standardinformation (wie z.B. Bildgröße, Bittiefe, etc.) verschlüsselt werden, da diese leicht geraten bzw. durch cut & paste von einem anderen File ersetzt werden können. Wesentlich für die Sicherheit ist demnach eine ausreichende Anzahl von Freiheitsgraden bei der Konfiguration des Kompressionsverfahrens die sich potentiell von Bild zu Bild unterscheiden können. Werden nicht nur Mainheaderdaten verschlüsselt, kann der Aufwand für die Identifikation der Headerdaten beträchtlich sein. Format compliance geht im allgemeinen natürlich verloren.

Beispiel: RTP Header Verschlüsselung I

Eine Idee ist nur einen Teil der Netzwerkprotokoll Header Daten zu verschlüsseln (formatunabhängig, siehe [2]), insbesondere wird vorgeschlagen nur einen Teil der RTP header Daten zu verschlüsseln. Im Gegensatz dazu lässt SRTP (RFC 3711) den Header unverändert sondern schützt nur die Payload.



Beispiel: RTP Header Verschlüsselung II



Idee: die timestamp und das CSRC Feld verschlüsseln. Die Annahme ist, dass durch das Vermischen der Pakete bei der Übertragung und die nicht-Verfügbarkeit der timestamp eine korrekte Anordnung nicht möglich ist. Jedoch ist die sequence number völlig ausreichend, um die Pakete in korrekte Reihenfolge zu bringen (das erste Paket bekommt eine zufällige Nummer, die im Anschluss inkrementiert wird. Das CSRC Feld ist überhaupt nur optional, daher ist die Annahme dass mehrere streams nicht getrennt werden können auch nicht korrekt. Selbst eine vollständige Verschlüsselung des RTP Headers ist noch unsicher, da die Annahme der “Selbstpermutation” durch Netzwerkübertragung nicht korrekt ist. Verschiedene RTP sessions können z.B. durch verschiedene UDP ports identifiziert werden, die Anordnung der RTP Pakete kann u.U. im IPv4 identification field rekonstruiert werden [35].

Beispiel: JPEG2000 Header Verschlüsselung I

Die Verschlüsselung des main headers kann normalerweise keine ausreichende Sicherheit bieten. Eine Alternative ist es, die packet header daten selektiv zu verschlüsseln. Motivation ist die geringe Menge an Header Daten verglichen mit den packet body Daten.

Cblk. Size	Layers	Header Bytes	Body Bytes	Ratio
64×64	16	1823	129072	1.4%
32×32	32	4603	126283	3.5%
16×16	32	11379	119510	8.5%
8×8	32	25748	105176	18.6%

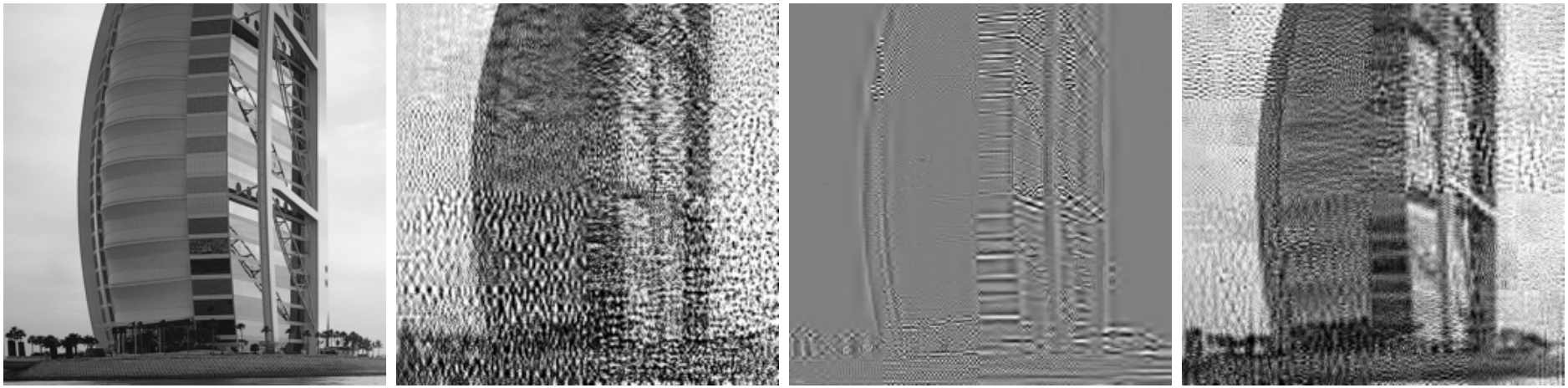
Es scheint klar, dass mit so einer Strategie keine confidentiality erreichbar ist, da die eigentlichen Bilddaten ungeschützt sind und die header die Information tragen, wie genau die Information zu interpretieren (i.e. dekodieren) ist. Daher ist das Zielszenario die transparente Verschlüsselung, wobei die format compliance ein zentraler Aspekt ist (um die Daten mit einem gewöhnlichen Viewer betrachten zu können).

Beispiel: JPEG2000 Header Verschlüsselung II

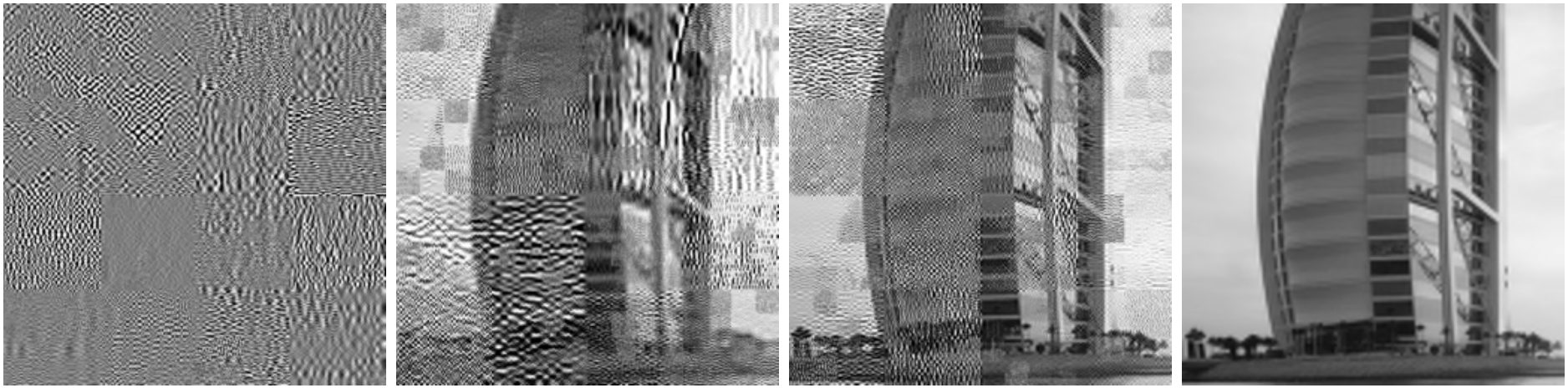
JPEG2000 packet header data besteht aus der Inklusions Information (daten welches codeblocks stehen im packet), der Länge der einzelnen codeblock Beiträge (CCP), der Anzahl der coding passes, und der Anzahl der Null-Bitplanes. Eine Nicht-Verfügbarkeit dieser Daten (bzw. Verwendung von falschen Daten) führt zu Fehlinterpretation der packet data in der Dekodierung. Für transparante Verschlüsselung bleiben die Header Daten am Beginn des Bitstroms (für das Preview-Bild) ungeschützt.

Die benötigte Transformation muss daher format compliant sein: es werden Permutationen benutzt, die spezielle header Eigenschaften aufrechterhalten müssen (z.B. muss die Gesamtlänge erhalten bleiben).

Beispiel: JPEG2000 Header Verschlüsselung III



Test image, CCP Längen und coding passes, Anzahl der leading zero bitplanes und inclusion information.



Transformationen auf Resolution 0, 2 und 3, sowie das Preview Bild (res. 2).

Partielle Verschlüsselung von Skalierbaren/Embedded Bitstreams

Bei der Organisation des Datenmaterials in Baselayer und mehrere Enhancementlayers (JPEG progressive Modi, MPEG-2, MPEG-4) bietet eine Verschlüsselung des Baselayers [10, 19] eine sehr effiziente Möglichkeit der partiellen Verschlüsselung. Im Fall von embedded Bitstreams (J2K, SPIHT) [4, 14, 20] kann die partielle Verschlüsselung noch feiner geregelt werden und der Beginn des Bitstreams bis zu einem beliebigen Truncation Point verschlüsselt werden.

Ein Spezielfall für diese Vorgangsweise ist die *transparente Verschlüsselung*, bei der potentiellen Kunden eine Version des Bildmaterials in geringer Qualität gratis zur Verfügung gestellt wird (Base layer [10, 19] oder der Beginn eines embedded Bitstreams [14]) um zum Kauf der Materials in voller Qualität anzuregen. Enhancement layers bzw. der hintere Teil des embedded Bitstreams werden verschlüsselt und können nur mit Hilfe des (zahlungspflichtigen) Schlüssels verwendet werden. Ziel ist also hier nicht Zeitreduktion oder Format compliance (beides kann aber realisiert werden), sondern zusätzliche Funktionalität.

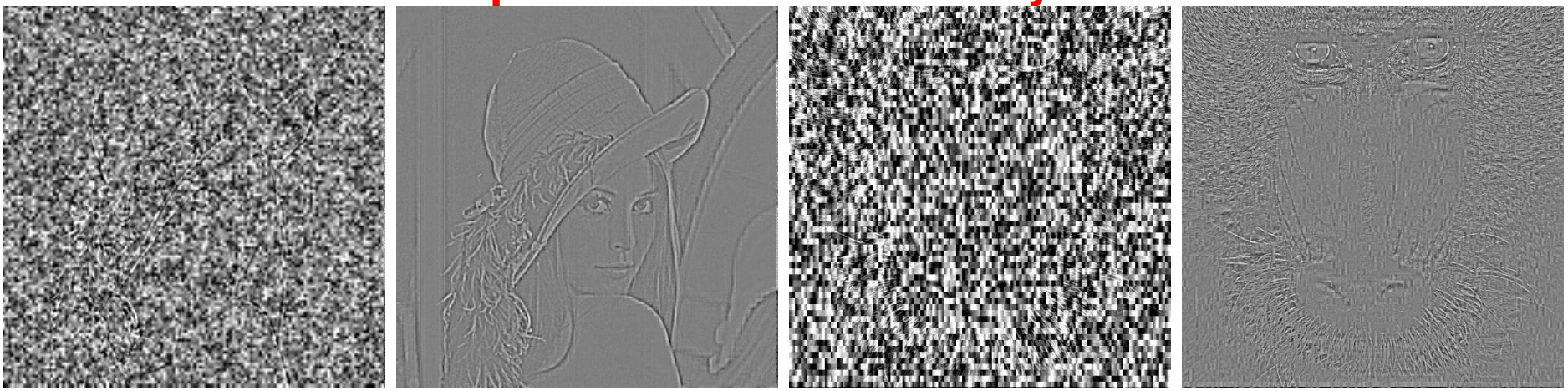
Beispiel: JPEG Extended System I

- Hierarchical progressive mode (HP): an image pyramid is constructed by repeated weighted averaging and downsampling. The lowest resolution approximation is stored as JPEG (i.e. the first scan), reconstructed, bilinearly upsampled, and the difference to the next resolution level is computed and stored as JPEG with different quantization strategy.

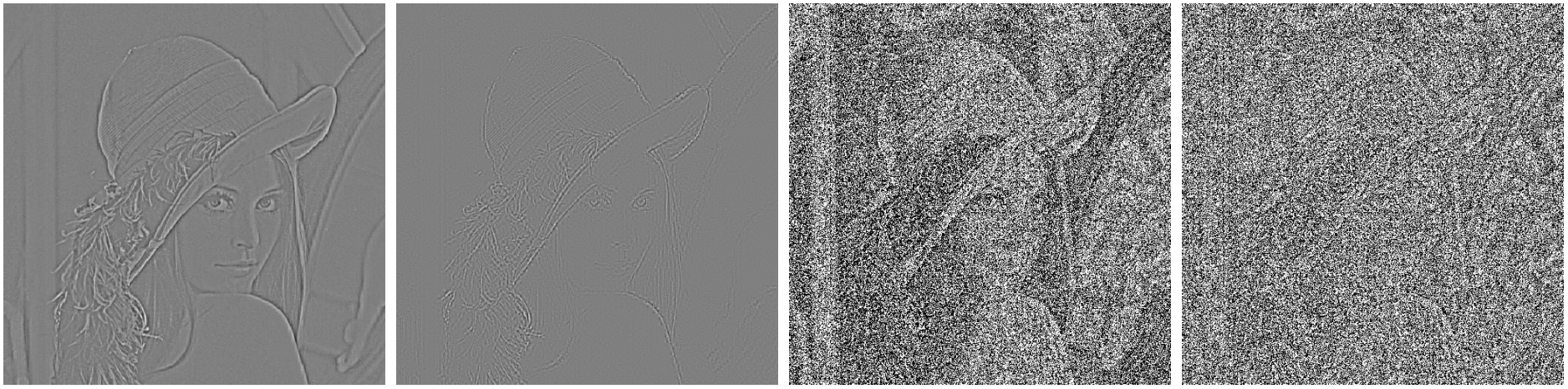
- Sequential progressive modes
  - ⋆ Spectral selection (SS): the first scan contains the DC coefficients from each block of the image, subsequent scans may consist of a varying number of AC coefficients, always taking an equal number from each block.
  - ⋆ Successive approximation (SA): the most significant bits of all coefficients are organized in the first scan, the second scan contains the next bit corresponding to the binary representation of the coefficients, and so on.

The mixed mode interleaves SS and SA.

Beispiel: JPEG Extended System II



3 level HP, lowest level encrypted (direct reconstruction and replacement attack); SS with DC and first AC coefficient encrypted.



HP, SS, SA, MM with 10% encrypted and reconstruction attack (uniform grayscale, zero coefficients, or 0-bitplanes replaced)

Bitstreambasierte Partielle Verschlüsselung: DCT I

Der naheliegenste Ansatz [3, 19] ist es, von jedem 8×8 Pixel Block das Datenmaterial (VLC Codewords), das den führenden Koeffizienten entspricht, mit einem Cipher zu verschlüsseln. Die Menge an zu verschlüsselnden Daten kann dadurch stark verringert und gut dosiert werden, der Aufwand ist aber erheblich diese Teile zu identifizieren. Außerdem geht im Allgemeinen trotz des Erhalts der Headerstrukturen die Format Compliance verloren (dazu später mehr). Die nicht geschützten Daten entsprechen einem high-pass gefilterten Bild und geben Kantenzüge und Texturen preis.

Eine Permutation des Bitstreams auf Byte Ebene mit einer fixen Permutationstabelle wird in [27] diskutiert (*Pure permutation Algorithm*). Die Länge der Permutationsliste (1 oder mehrere Byte) kann der Sicherheitserfordernis angepasst werden. Wie alle Permutationsvefahren kann dieses Verfahren durch eine known Plaintext Attacke leicht gebrochen werden. Die Geschwindigkeit ist allerdings sehr hoch.

Bitstreambasierte Partielle Verschlüsselung: DCT II

Eine eingehende Untersuchung der statistischen Eigenschaften von DCT-basieren Daten auf Byte Ebene motiviert die Entwicklung des *Video Encryption Algorithm* [27]. Bytes werden aus folgenden Gründen als geeignete Verarbeitungseinheit betrachtet:

- Byte Operationen lassen sich effizient realisieren.
- Ein einzelnes Byte hat keine Bedeutung im Kontext eines Bitstreams (VLC Codewords meist mehrere Bytes)
- Durch die Huffman VLC Codewords wird eine hohe Entropie auf Byteebene verursacht.

Die Bytes im Bitstream sind nahezu gleichverteilt, das gilt auch für Paare von Bytes und für beliebige Teile des Bitstreams. Ebenso kommen keine Musterwiederholungen vor (wird in [27] experimentell belegt).

Funktionsweise: Man bildet zwei neue Byte Streams (Oddlist – alle Bytes ungerader Position, Evenlist analog) und XORed diese. Das Ergebnis dieser Operation concatiniert mit der DES verschlüsselten Evenlist bildet den Ciphertext. Die verschlüsselte Evenlist ist in diesem Fall ein one-time Pad.

Bitstreambasierte Partielle Verschlüsselung: DCT III

Um die Sicherheit des *Video Encryption Algorithm* zu erhöhen wird ein zusätzlicher Schlüssel vorgeschlagen, der die Auswahl der Bytes für die beiden Listen regelt. Bei regelmäßiger Generierung der Listen kann nämlich bei bekanntem Plaintext durch XOR die verschlüsselte Evenlist erzeugt werden. Dieser Schlüssel wird für jeden Frame gewechselt. Zusätzlich wird die Evenlist mit 8 verschiedenen Permutationslisten (eine für jeweils 32 Bytes) bearbeitet.

Die Sicherheit des Verfahrens ist sehr gut. Allerdings ergibt sich eine Komplexitätsreduktion von nur 47% des reinen Verschlüsselungsaufwands verglichen mit vollständiger DES Verschlüsselung. Die Bitstreamstruktur wird zerstört, das Verfahren ist also nicht format compliant ! Aufwendiges Schlüsselmanagement (Keys für DES, die Listengenerierung und die Permutationen). Betrachtet man den geringen Gewinn, scheint das Verfahren wenig praktikabel.

Bitstreambasierte Partielle Verschlüsselung: DCT IV

Um zwei format compliant Verfahren für das MPEG-4 Intellectual Property Management and Protection (IPMP) System zu motivieren, werden von Wen et al. [42] zwei weitere Probleme bei nicht standardkonformen Bitstreams angeführt.

Auch wenn Headerstruktur erhalten bleibt, können durch die nicht format compliant Verschlüsselung der Datenfelder Marker und Header emuliert werden, die eine korrekte Interpretation des Bitstreams unmöglich machen. Die Verschlüsselung einer Concatinierung von VLC Codeword Bits führt in Allgemeinen nicht zu einer gültigen Concatinierung von VLC Codewords ! Beispiel: gegeben seien Codewords 0, 10, 110, 111 und eine Concatinierung 010. Die Verschlüsselung kann z.B. zum Ergebnis 001 führen, was keine gültige Codeword Concatinierung ist.

Die beiden von Wen et al. [41, 42] vorgeschlagenen Verfahren lösen zwei Probleme früherer Ansätze:

1. Erhalt gültiger VLC Codewords trotz Einsatzes starker Kryptographie (im Gegensatz etwa zu [19]).

2. Erhöhung der Sicherheit bei VLC Codeword Permutationen.

Bitstreambasierte Partielle Verschlüsselung: DCT V

Das beschriebene Verfahren zur Verschlüsselung von VLC Codewords arbeitet für Tabellen mit $N = 2^n$ VLC Codewords. Ist das nicht erfüllt, wird N durch mehrere kleine Tabellen mit dieser Eigenschaft konstruiert.

Vor der Verschlüsselung wird jedem VLC Codeword ein n-Bit Index fester Länge in einer Tabelle zugewiesen. Die zu verschlüsselnden Codewords werden dann zu C concatiniert und der entsprechende Indexstring S erzeugt. S wird nun mit einem Cipher verschlüsselt zu S'. S' wird nun über die zuvor definierte Tabelle wieder in eine Concatinierung von VLC Codewords C' rückabgebildet. Diese werden an die ursprüngliche Position von C im Bitstream geschrieben.

Wichtig ist natürlich daß der verwendete Cipher die Anzahl der Bits konstant läßt, weil sonst nach der Verschlüsselung mehr VLC Codewords vorhanden sind. C und C' werden sich im Allgemeinen in ihrer Bitanzahl unterscheiden. Im Gegensatz zu VLC Codewords können FLC Codewords direkt verschlüsselt werden (z.B. DCT sign Bits).

Bitstreambasierte Partielle Verschlüsselung: DCT VI

Im Gegensatz zu Koeffizientenpermutation [36, 31] und bytewieser Permutation [27] sollen semantische Einheiten des Bitstreams permutiert werden. Dafür bieten sich insbesondere zwei Möglichkeiten an:

1. Ganze 8×8 Pixel Blöcke werden permutiert: Hier gibt es kaum etwas zu beachten außer daß die Anzahl der gemeinsam verarbeiteten Blöcke groß genug sein muß. Eine Variante ist die DC Koeffizienten stehen zu lassen und diese mit einem Cipher zu verschlüsseln (sind FLC DPCM codiert !).

2. VLC Codewords werden permutiert: wie von Zeng und Lei [44] bereits vorgeschlagen, können Codewords je eines Frequenzbereiches über Blockgrenzen hinaus mit eigenen Tabellen permutiert werden. Auf die unterschiedliche Anzahl von Codewörten in den Blöcken muß geachtet werden, insbesondere muß bei Involvierung des letzten Codewords im Block auf eine Umsetzung des “last” Feldes geachtet werden.

Bitstreambasierte Partielle Verschlüsselung: DCT VII

Um eine Erhöhung der Sicherheit der Permutationen gegen known Plaintext Attacken zu erreichen, wird eine verschlüsselungs-basierte, on-the-fly Generierung der Permutationstabellen vorgeschlagen. Um das nicht durch erhöhten Aufwand des Keymanagements realisieren zu müssen (vgl. Nagravision oder Videocrypt), wird eine Generierung der Tabellen aus “loakalen” Daten (also Teilen des Bitstreams die nicht in die Permutation involviert sind) realisiert. Im Fall der VLC Codeword Permutation können dafür z.B. DES verschlüsselte DCT Sign Bits verwendet werden, die durch ihre hohe Entropie nicht mitpermutiert werden müssen.

Bitstreambasierte Partielle Verschlüsselung: DCT VIII

Beispiel für die Generierung einer Permutationstabelle:

- DCT Sign Bits und DC Information wird DES verschlüsselt (Key K_F).
- Eine zufällige Bitfolge R_L der Länge L wird generiert durch einen Cipher und dem Schlüssel K_L (fest für Bild). $L > bitlength \times K$ für alle $bitlength \times K$, wobei K die Anzahl der Codewords in einer Codeworttabelle und $bitlength$ im wesentlichen $log_2(K)$ ist.
- Für jede zu permutierende Codeword Menge (Codewords die untereinander permutiert werden) werden die verschlüsselten Sign Bits zu R' concatiniert.
- R' wird nun mit dem Key K_T verschlüsselt und ergibt den Output R, der $bitlength$ Mal wiederholt wird, was Rr ergibt.
- $Rc = R_L \ \ XOR \ \ Rr$.
- Rc wird in K nicht überlappende Segmente geteilt, jedes mit $bitlength$ Bits. Die Permutationstabelle bildet jeden Indexinputwert i (von 0 bis $K-1$) auf das i-te Segment von Rc ab.

Bitstreambasierte Partielle Verschlüsselung: DCT IX

Beurteilung der in [41, 42] vorgestellten Verfahren:

- Es kann partielle Verschlüsselung zur Erhaltung der format compliance aber auch zusätzlich zur Komplexitätsreduktion durchgeführt werden (durch Schützen einer Codeword Teilmenge entsprechend den Vorschlägen anderer Arbeiten).
- Die Resistenz der Permutation gegen known Plaintextattacken wird erreicht, die Anfälligkeit gegen Ciphertext only Attacken (durch bekannte Größenanordnung der Koeffizienten und der bekannten VLC Tabellen) bleibt jedoch erhalten.
- Der Verarbeitungsoverhead insbesondere bei der Verschlüsselung von VLC Codewords über Indextabellen ist beträchtlich, ebenso für die Verschlüsselung der Sign Bit Folgen zur Konstruktion der Permutationstabellen.
- Im Moment sicher der am weitesten fortgeschrittene Ansatz in der DCT Domain der viele Ideen anderer Systeme aufgreift und wesentlich verbessert.

Partielle Bildverschlüsselung im Kompressionsprozeß: Wavelets I

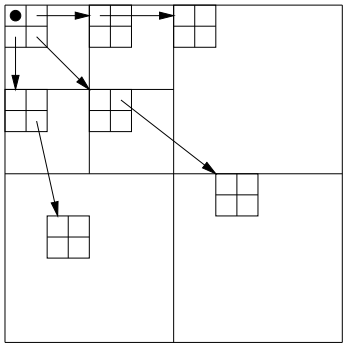
In Analogie zur DCT Domain werden hier ebenfalls zwei Ansätze zur Koeffizeintenmanipulation verfolgt, die jedoch eine etwas andere Beurteilung als im DCT Fall nach sich ziehen:

1. Verschlüsselung von Sign Bits von Koeffizienten [44, 14]: da auch die Sign Bits der WT hohe Entropie aufweisen und dadurch kaum komprimierbar sind, eignen sie sich gut für Verschlüsselung. Durch die spatiale Lokalisation der Koeffizienten bleiben Umrisse von Objekten schemenhaft erhalten. Format compliance bleibt erhalten.

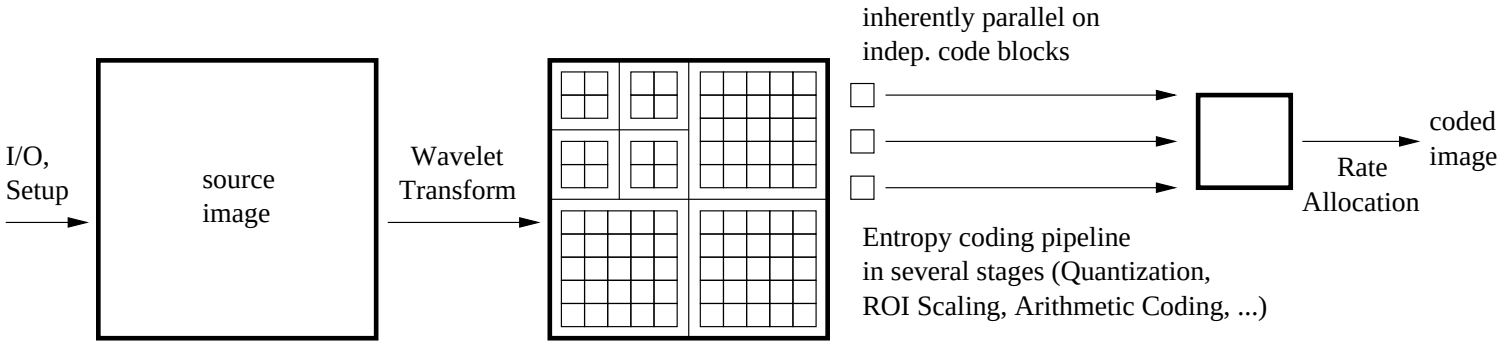
2. Permutation von Koeffizienten [44, 39]: Die Anfälligkeit gegenüber der known Plaintext Attacke ist bei fixen Permutationstabellen (durchaus unterschiedlich pro Subband) ebenfalls gegeben. [39] schlägt als zusätzliche Sicherung eine Verschlüsselung des Low-Pass Subbands vor. Dadurch geht aber eventuell format compliance verloren. [44] kombiniert die Permutation mit zusätzlicher Block Rotation und Sign Bit Verschlüsselung. Durch die bildabhängige spatiale Lokalisierung der Koeffizienten ist eine Ciphertext only Attacke im Gegensatz zum DCT Fall nicht möglich. Ebenso tritt der Abfall der Kompressionsleistung (je nach Verfahren) moderater auf. Kontextbasierte Verfahren (wie J2K) oder zerotree basierte (wie SPIHT oder EZW) lassen jedoch einen deutlichen Abfall erwarten.

Beispiel: Permutationen im Wavelet Transformationsbereich I

Whereas **S**et **P**artitioning **I**n **H**ierarchical **T**rees relies on spatial orientation trees (S.O.T. – zerotree like structures) thereby using inter **and** intra subband correlations, JPEG2000 coding is based on codeblocks which limits its scope to intra subband correlation.



S.O.T.



J2K Coding pipeline

Beispiel: Permutationen im Wavelet Transformationsbereich II

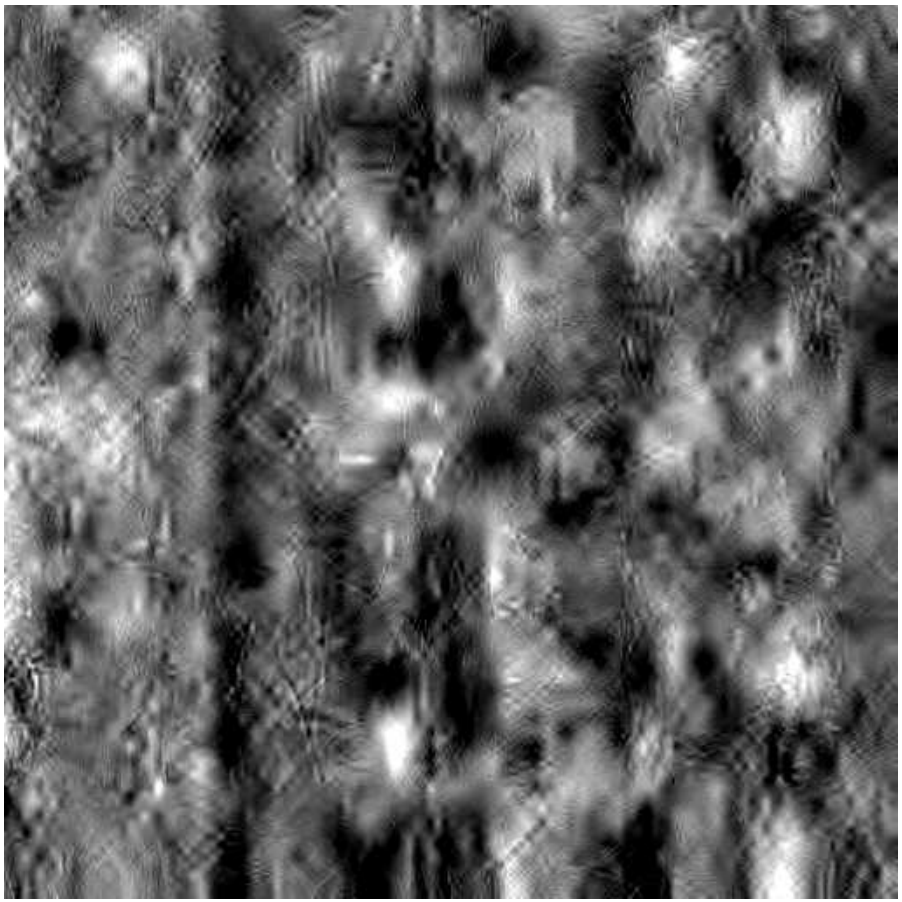
The following permutation scenarios are considered:

1. **blockwise-fixed**: Blocks of fixed size across all subbands are used ($1 \times 1, 2 \times 2, 4 \times 4, 8 \times 8,$ or 16×16 coefficients).

2. **blockwise-adaptive**: Each wavelet-subband is divided into 64 equally sized blocks.

   - arbitrary permutations across the subbands
   - identical permutations across all subbands: here entire multiscale trees are permuted as a whole !!

Beispiel: Permutationen im Wavelet Transformationsbereich III



Arbitrary Permutations

Identical Permutations

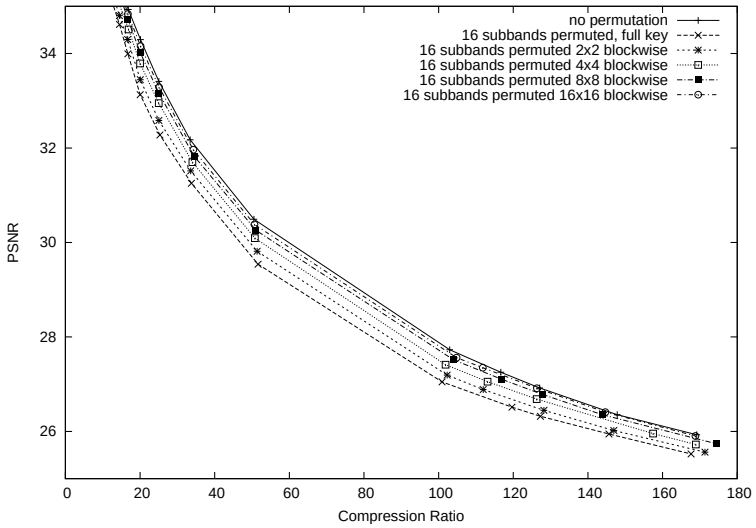
Beispiel: Permutationen im Wavelet Transformationsbereich IV

Each testimage is encoded with both considered coding algorithms. Within the coding pipeline, the coefficients of the different wavelet subbands are permuted before the quantization stage using one of the proposed permutation variants. The filesize (i.e. compression ratio) is recorded.

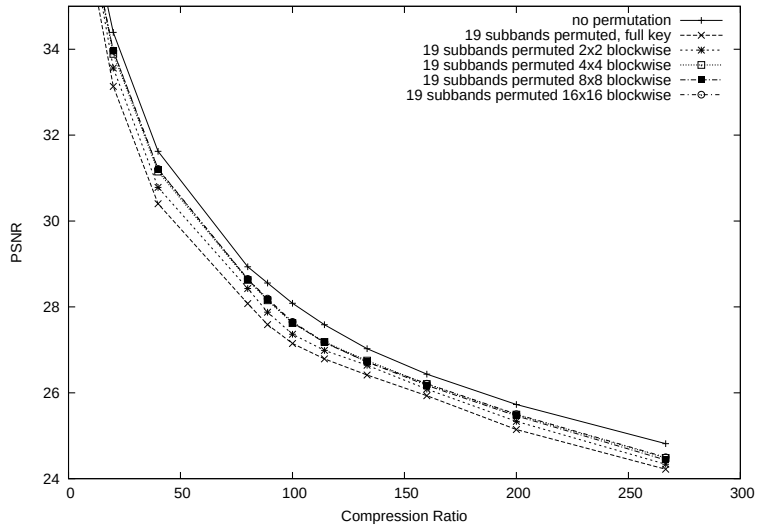
Thereafter, the encrypted and compressed file is decoded and the corresponding wavelet-subbands inversely permuted ('decrypted'). The image qulaity is recorded. Finally the overall rate versus distortion performance is computed.

Additionally, the efficiency loss measured in percentage of filesize increase as compared to the original coder is computed.

Beispiel: Permutationen im Wavelet Transformationsbereich V



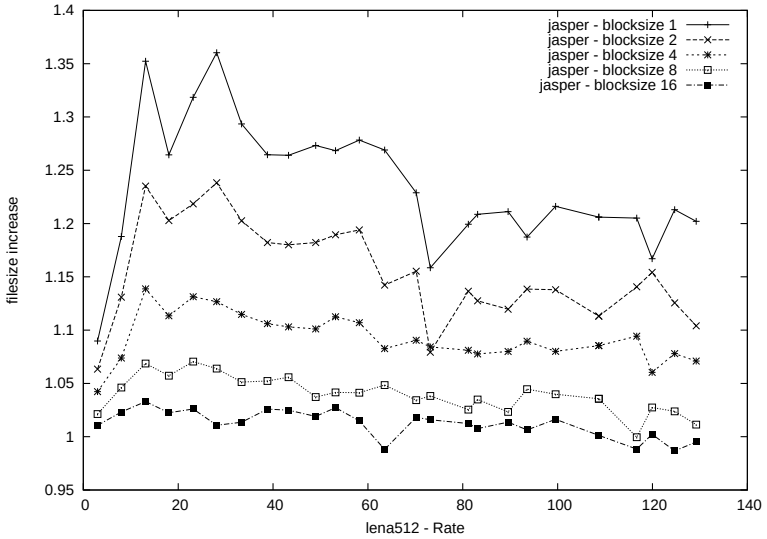
JPEG2000, blockwise-fixed



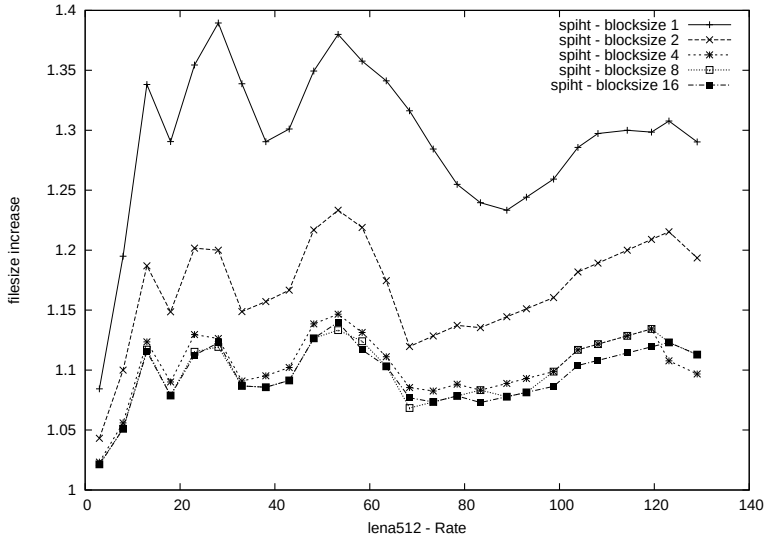
SPIHT, blockwise-fixed

In this representation, differences are hardly visible !

Beispiel: Permutationen im Wavelet Transformationsbereich VI



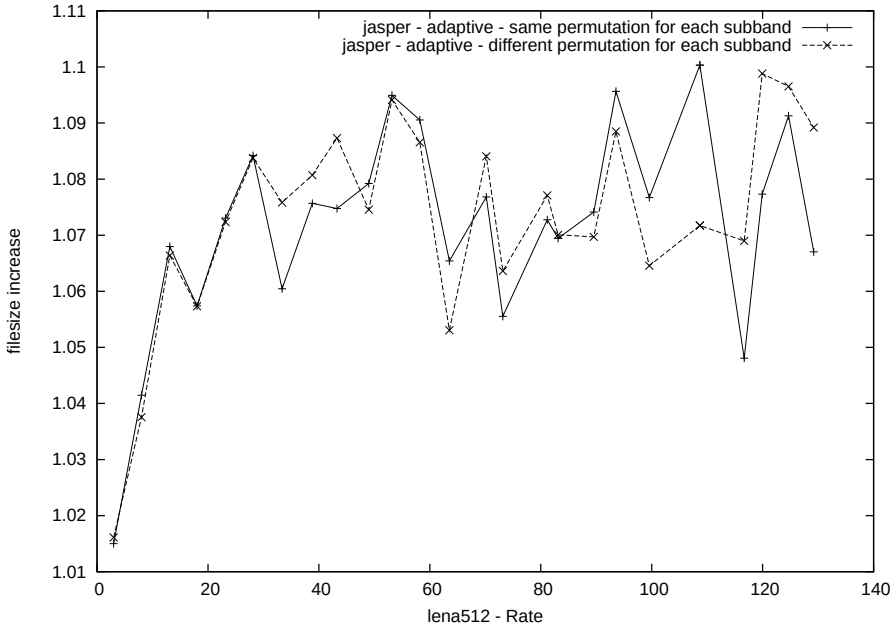
lena512, JPEG2000, blockwise-fixed



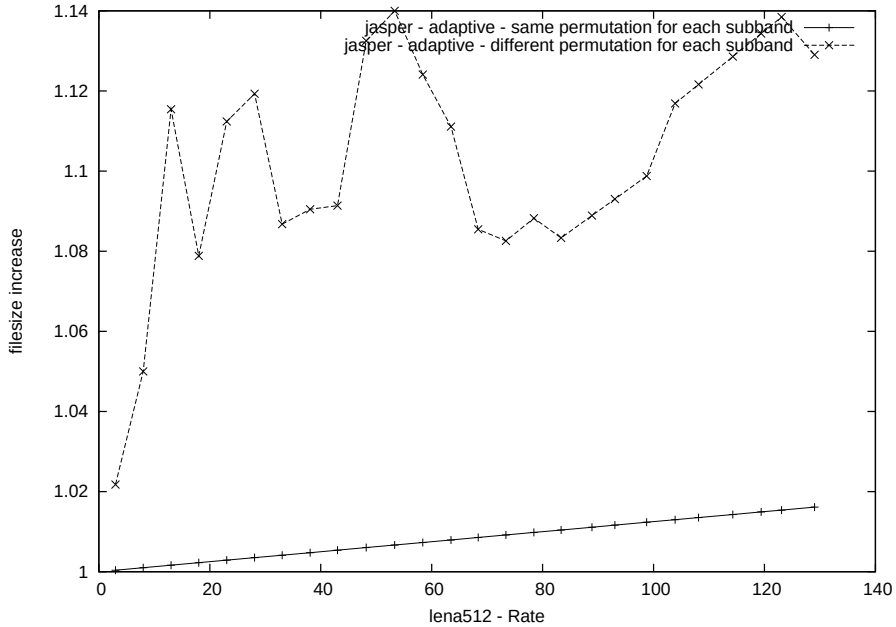
lena512, SPIHT, blockwise-fixed

There are subtle differences between the results of JPEG2000 and SPIHT. In the JPEG2000 case increasing the blocksize improves the result steadily up to blocksize 16×16, but we notice a saturation of the improvement at a blocksize of 4×4 pixels for SPIHT. Contrasting to JPEG2000, SPIHT cannot take advantage of a larger blocksize.

Beispiel: Permutationen im Wavelet Transformationsbereich VII



JPEG2000



SPIHT

We clearly note the different behaviour of SPIHT vs. JPEG2000: in the SPIHT case, using identical permutations accross the subbands does not harm the compression efficiency whereas it does not make a difference for JPEG2000. The inter subband correlations as preserved contribute about 10% of the file size in the SPIHT case.

Partielle Bildverschlüsselung im Kompressionsprozeß: Wavelets II

Durch die Verfügbarkeit einer großen Variabilität von Wavelet ähnlichen Transformationen lassen sich in diesem Bereich spezielle Versionen von Headerverschlüsselung realisieren, die auf der Verwendung von “geheimen” Transformationsdomains beruhen. Idee ist hier, daß es dabei genügt, die im Header spezifizierte Art der Transformation zu schützen. Minimaler Kompressionsaufwand und zumindest teilweise format compliance (Bitstream wird nicht geschützt) sind die Folge.

Verwendete Transformationen sind:

- Wavelet Packets [24, 25, 26] verwenden geheime Subbandstrukturen.
- NSMRA: Verwendung von verschiedenen (geheimen) Wavelet Filtern auf verschiedenen Zerlegungsstufen [22].
- Parametrisierte Wavelets: aus einem großen Parameterraum werden geheim bestimmte Waveletfunktionen gewählt.

Hauptprobleme dieser Verfahren sind die Kontrolle der Kompressionsqualität verschiedener Transformationen und eventuell mögliche Rückschlüsse auf die verwendete Transformation durch Analyse der Transformationskoeffizienten.

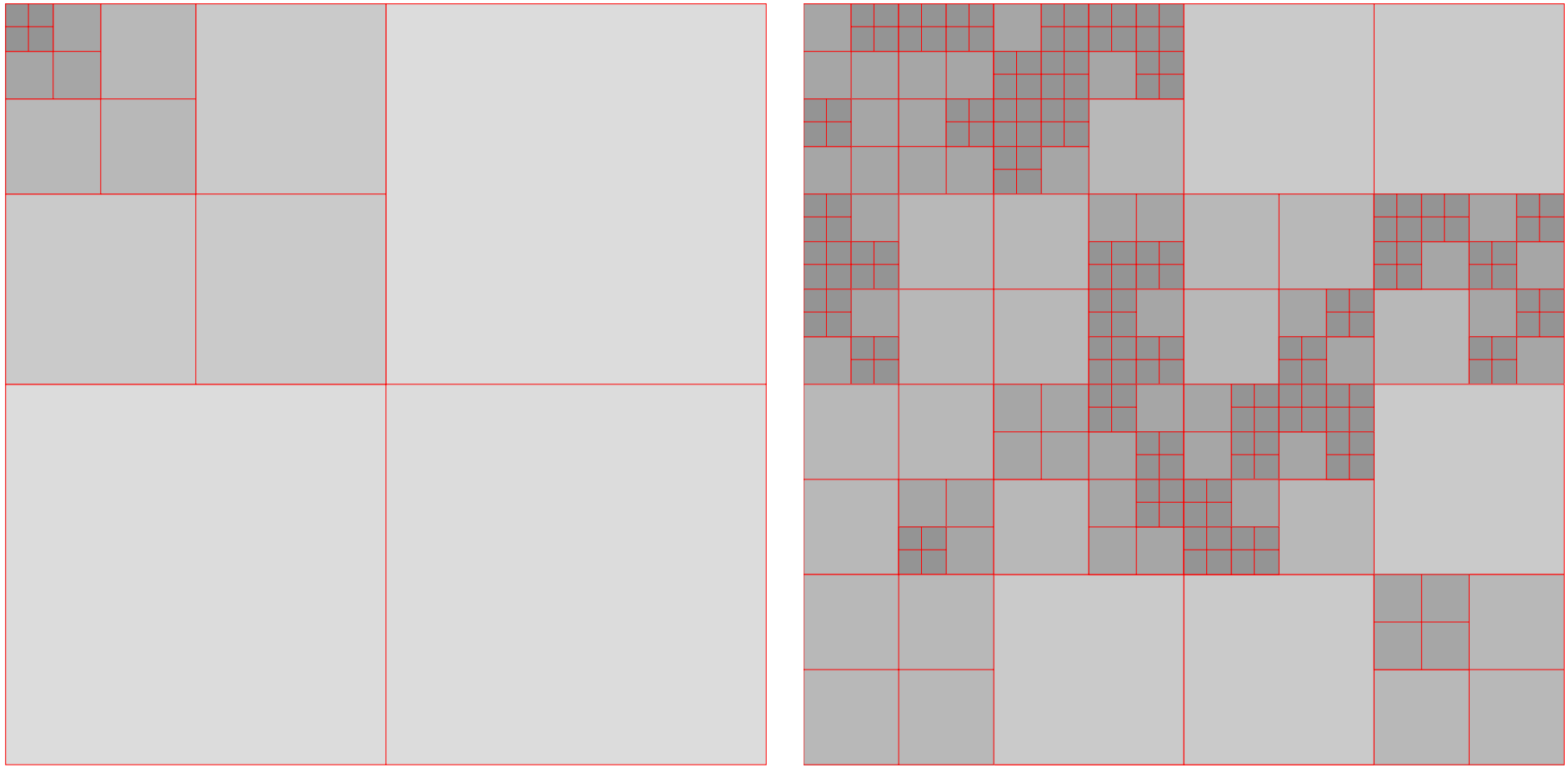
Partielle Bildverschlüsselung durch geheime Wavelet Domains: Wavelet Packets I

Wavelet Packets sind eine Verallgemeinerung der pyramidalen Wavelet Transformation, bei der nicht nur das Low-Pass Subband rekursiv weiterzerlegt wird, sondern auch alle Detail Frequenzbänder. Die Folge ist eine wesentlich bessere Freuqenzauflösung insbesondere der höheren Frequenzbereiche. Im 2-D Fall ensteht durch die Zerlegung ein vollständiger Quadtree, aus dem diejenigen Teile ausgewählt werden, die ein Bild gut darstellen.

Anwendungsbereiche für Wavelet Packets:

- Kompression: Best Basis Algorithmen mit Kostenfunktionen, FBI Fingerprint Standard, geeignet v.a. für Texturen mit hohen Frequenzanteilen
- Signalklassifikation, wieder insbesondere für Texturen
- Numerische Mathematik für effiziente Darstellung von Operatoren

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Wavelet Packets II



(a) Standard wavelet decomposition

(b) Typical wavelet packet decomposition

Abbildung 5: Decomposition trees for different subband structures

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Wavelet Packets III

Durch einen Zufallszahlengenerator wird eine Quadtreestruktur erzeugt, die die Wavelet Packet Subbandstruktur festlegt. Anschließend wird das Bild entsprechend zerlegt und komprimiert. Nur der Seed des baumerzeugenden Zufallsgenerators muß verschlüsselt gespeichert werden, der Rest der Daten (also die gesamten verarbeiteten Koeffizienten) bleibt im Plaintext.

Die Erzeugung der Quadtreestruktur geschieht durch Zufallsentscheidung für jedes Subband, ob es weiterzerlegt wird oder nicht. Eine Subbandstruktur mit Zerlegungstiefe 1 hat daher Auftrittswahrscheinlichkeit $1/2$ (Bild zerlegt oder nicht), mit Zerlegungstiefe 5 allerdings nur mehr $\frac{1}{2^5}$. Flache Zerlegungen sind also wahrscheinlicher als tiefe – deshalb werden bei der Baumerzeugung zusätzliche Gewichte verwendet, die das korrigieren.

Eine wichtige Frage ist nun, welche Kompressionsqualität solch zufällig erzeugten Subbandstrukturen liefern und ob bestimmte wegen geringer Qualität ausgeschlossen werden müssen. Es stellt sich heraus daß man eine Mindestzerlegungstiefe für das Low-Pass Subband festsetzen muß.

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Wavelet Packets IV

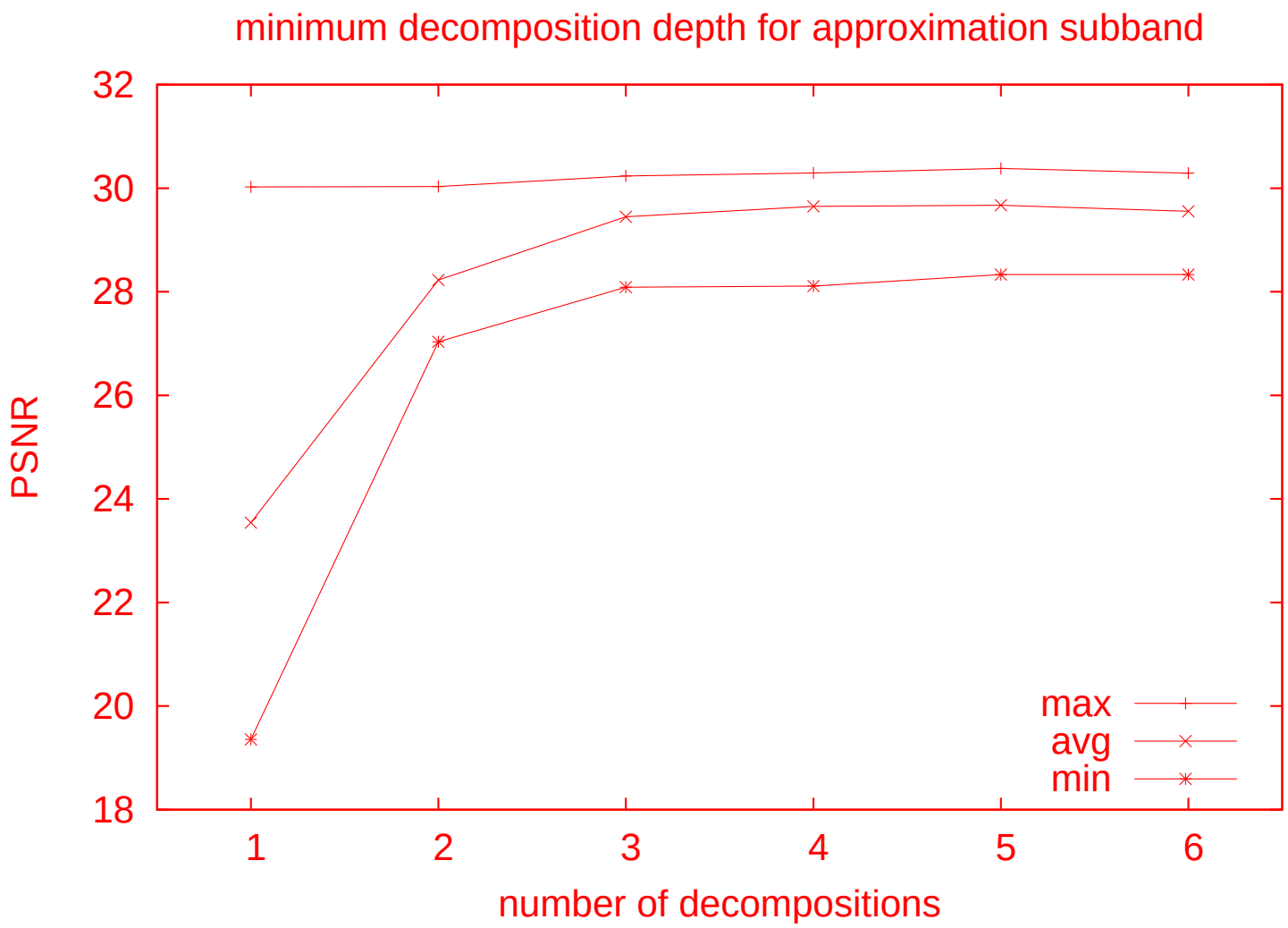


Abbildung 6: approximation subband minimum decomposition depth

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Wavelet Packets V

Mögliche Angriffe gegen das Verfahren:

- Brechen der Verschlüsselung des PRNG Seeds: Unrealistisch (z.B. AES Verschlüsselung)
- Brute Force Attacke gegen Seed: vernünftiger PRNG erlaubt das nicht.
- Rekonstruktion der Subbandstruktur mit Hilfe der unverschlüsselten Daten: wird im Folgenden besprochen.
- Vollständiges Durchtesten aller möglichen Subbandstrukturen: bei einer Beschränkung der maximalen Zerlegungsanzahl auf 6 (auch aus Qualitätsgründen sinnvoll) ergeben sich 2^{4185} verschiedene Subbandstrukturen. Unrealistisch.

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Wavelet Packets VI

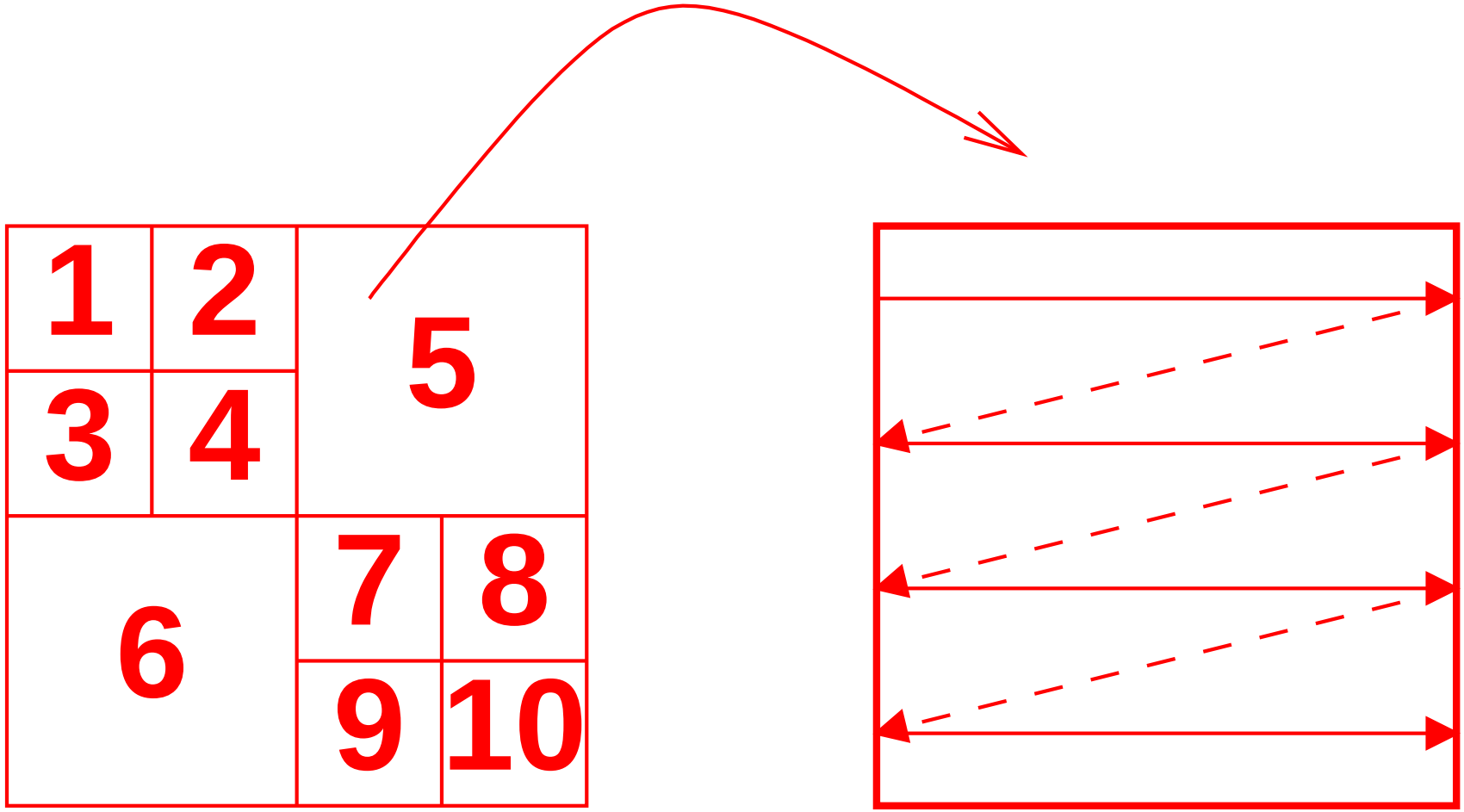


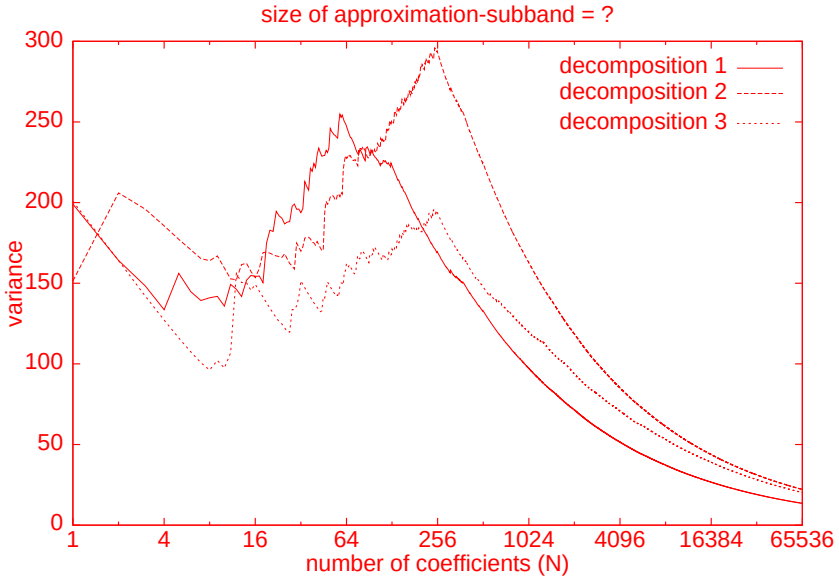
Abbildung 7: Example for our scan order assumption

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Wavelet Packets VII

Der erste (und wichtigste) Schritt in einer Attacke die die Subbandstruktur zu rekonstruieren versucht ist es die Koeffizienten des Low-Pass Subbands zu identifizieren. Die statistischen Eigenschaften dieses Subbands sind wesentlich anders als die der Detail Subbands, insbesondere sind die Koeffizientenwerte in etwas gleichverteilt, im Gegensatz zur starken Häufung um 0 bei den anderen Subbands. Diese Eigenschaft kann ausgenutzt werden, z.B. durch die Berechnung einer modifizierten Varianz $v = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - m)^2$, wobei m der Mittelwert aller Koeffizienten ist und N ist die Anzahl der untersuchten Koeffizienten. v wird für wachsende N evaluiert. Der scharfe Abfall in der Kurve (nächste Folie) zeigt die Anzahl der Koeffizienten an.

Die Größe von Detail Subbands kann leicht durch die Korrelation von Subbandzeilen bestimmt werden, da hier eine ähnliche Eigenschaft wie für Bilder gilt. Diese Art von Attacke kann bei Zerotree basierten Kodern nicht durchgeführt werde, da die komprimierten Daten weder subbandorientiert noch koeffizientenweise gespeichert werden.

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Wavelet Packets VIII



(a) Variance for increasing number of coefficients

(b) Reconstruction using a wrong decomposition tree but the correct approximation subband size

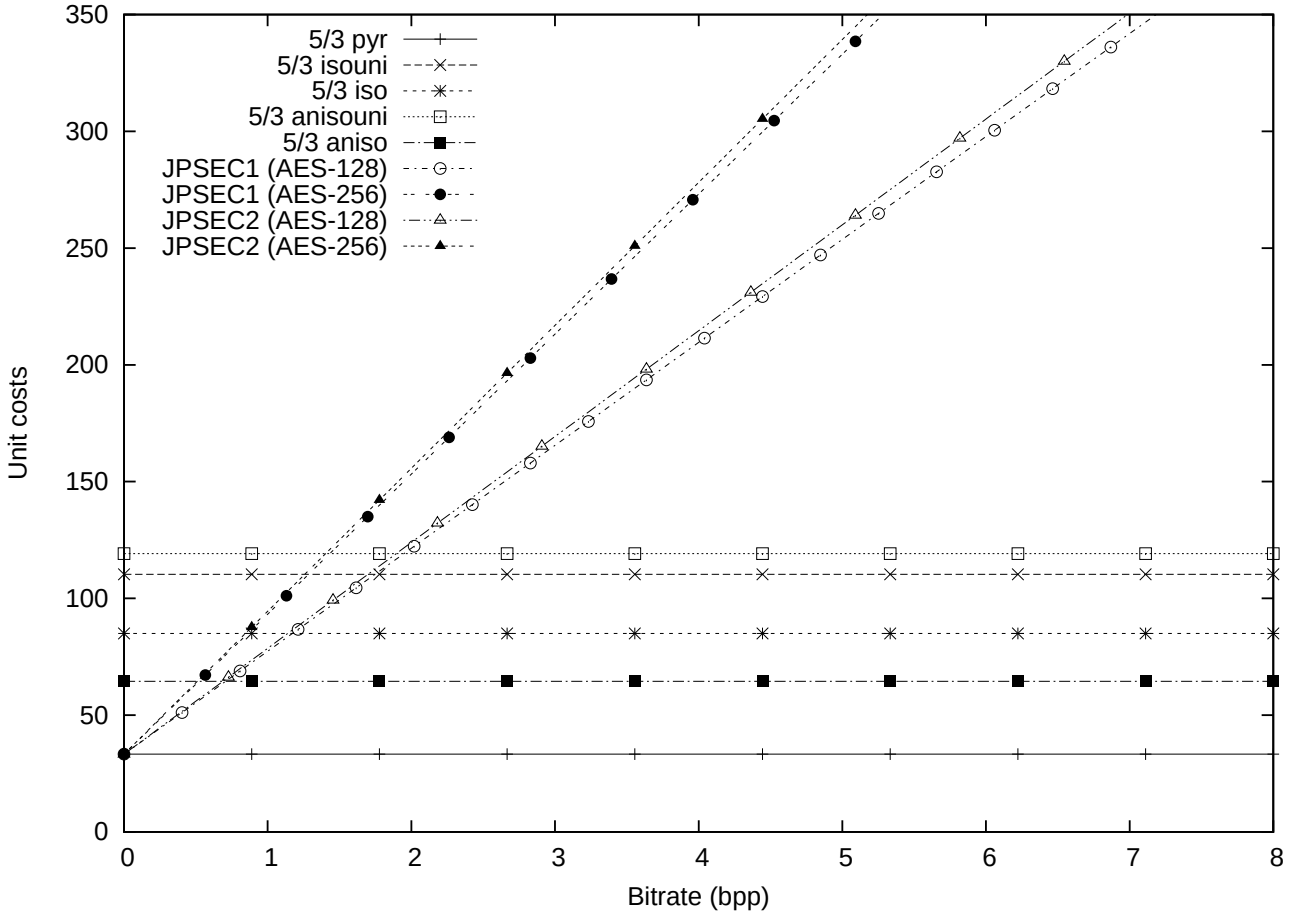
Partielle Bildverschlüsselung durch geheime Wavelet Domains: Wavelet Packets IX

Die Hauptmotivation für den Ansatz liegt in der extrem geringen Datenmenge die verschlüsselt werden muss, was der Grund ist diese Systeme als “lightweight” zu klassifizieren. Im Folgenden soll die Gesamt-Komplexität abgeschätzt werden, die für folgende Systeme benötigt wird:

- JPSEC1: Klassische AES Verschlüsselung des J2K Bitstroms
- JPSEC2: Format compliant AES Verschlüsselung des J2K Bitstroms (i.e. packet header preservation, no marker emulation)
- isouni: WP basierte Verschlüsselung ohne Kompressionsberücksichtigung
- iso: WP basierte Verschlüsselung mit Kompressionsberücksichtigung

Um für die WP-basierten Verfahren die Komplexität abschätzen zu können, muss die durchschnittliche Zerlegungstiefe die durch die “zufällinge” Erzeugung entsteht, berechnet werden !

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Wavelet Packets X



Partielle Bildverschlüsselung durch geheime Wavelet Domains: Wavelet Packets XI

Die Abschätzung bisher beruht auf einem idealisierten Rechenmodell mit Einheitskosten für Operationen und Speicherzugriff. Tatsächliches Laufzeitverhalten für Kompression und Verschlüsselung stellt sich wie folgt dar:

structure	compression		decompression	
pyramidal	0.643 s	1.55 fps	0.404 s	2.47 fps
iso	1.005 s	1 fps	0.962 s	1.04 fps
isouni	1.147 s	0.87 fps	1.011 s	0.99 fps

JPSEC encryption			
Method	throughput	codestreams with 2bpp	
JPSEC1	42.71 MB/s	0.0015 s	683.36 fps
JPSEC2	37.81 MB/s	0.0017 s	604.96 fps

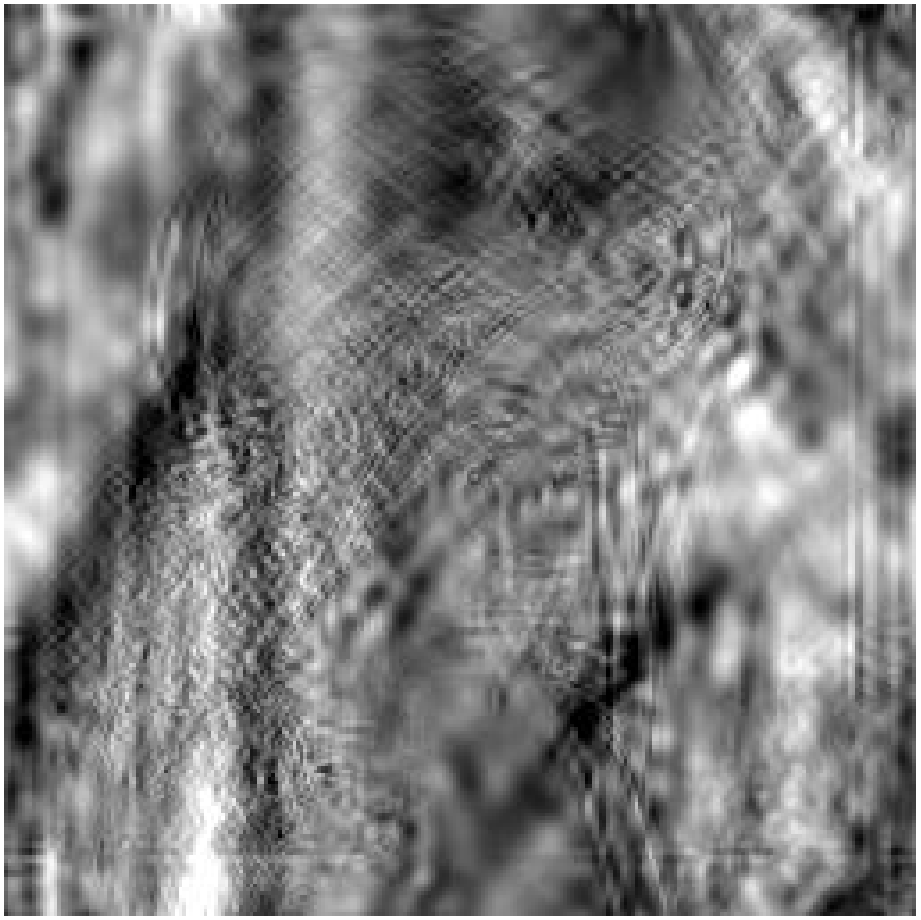
Insgesamt dominiert also bei weitem die Kompressionszeit, daher kann die signifikante Laufzeitsteigerung durch WP NICHT durch die geringe Verschlüsslung wettgemacht werden. BEMERKUNG: das gilt für gegenwärtige Software und Hardwareverhältnisse (wenig optimierte Codecs vs. voll optimiertes AES in C, Cache Probleme in der Wavelettransformation).

Partielle Bildverschlüsselung durch geheime Wavelet Domains: NSMRA

Non-stationary Multiresolution Analyses (NSMRA) sind pyramidale Wavelet Zerlegungen die für die verschiedenen Zerlegungsebenen unterschiedliche Filter verwenden. Analoger Ansatz zu Wavelet Packets: nur die Information welche Filter verwendet wurden, wird verschlüsselt, der Rest bleibt im Plaintext. Für das Verfahren in [22] benötigt man eine Library mit l Filtern und Zerlegungstiefe k. Das ergibt einen Schlüsselraum der Größe l^k, die für die Kompression/Verschlüsselung tatsächlich verwendeten Filter werden zufällig ausgewählt.

Ein brute force Angriff ist bei vernünftiger Größe der Filterlibrary kaum möglich, jedoch senkt folgende Attacke die Anzahl der zu testenden Filter auf $l \times k$: wird der korrekte Filter verwendet, ergibt sich ein Bild/Subband mit geringerem Rauschen als mit einem falschen. Die Entropie der Differenz von benachbarten Pixeln/Koeffizienten dient als Maß für Glattheit. Nun wird, beginnend mit der ersten Zerlegungsstufe, Stufe für Stufe die Zerlegung durchgegangen und in jedem Schritt der beste Filter für eine Stufe fixiert. Durch die Existenz dieser Attacke und den eher kleinen Schlüsselraum eher für low-Security Applikationen verwendbar.

Partielle Bildverschlüsselung durch geheime Wavelet Domains: NSMRA Beispiele



(c) Reconstruction using (wrong) random filters

(d) Reconstructed image where the heuristic failed at 3 out of 5 levels

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Parametrisierte Filter

Grundidee ist, aus einer großen Familie von parametrisierten Wavelet Filtern durch zufällige Parameterwahl einen Filter auszuwählen und die Kompression damit durchzuführen. Wieder wird nur der/die Parameter verschlüsselt und die eigentlichen Bitstreamdaten bleiben ungeschützt. Dabei gibt es folgende Probleme:

- Die durch die Parametrisierung erhaltenen Filter haben unterschiedliche Qualität die sich nicht so leicht steuern läßt wie im Wavelet Packet Fall. Unklar ist, ob es genügend gute gibt für einen sinnvoll großen Keyspace. Vorgehensweise: Filter generieren, Kompression durchführen, bei zu schlechter Qualität verwerfen und von neuem beginnen. Ineffizient !

- Sinnvolle Parametrisierungen mit ausreichender Anzahl von Filtern relativ guter Qualität sind nur für den orthogonalen Fall verfügbar, der klar schlechter komprimiert !

- Filter mit ähnlichen Parameterwerten komprimieren ähnlich, was den Keyspace einschränkt und ein Sicherheitsproblem darstellt.

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Parametrisierte Filter

Dilation Equation für orthogonale Wavelets mit kompaktem Träger:

$$\phi(t) = \sum_{k \in Z} c_k \phi(2t - k),$$

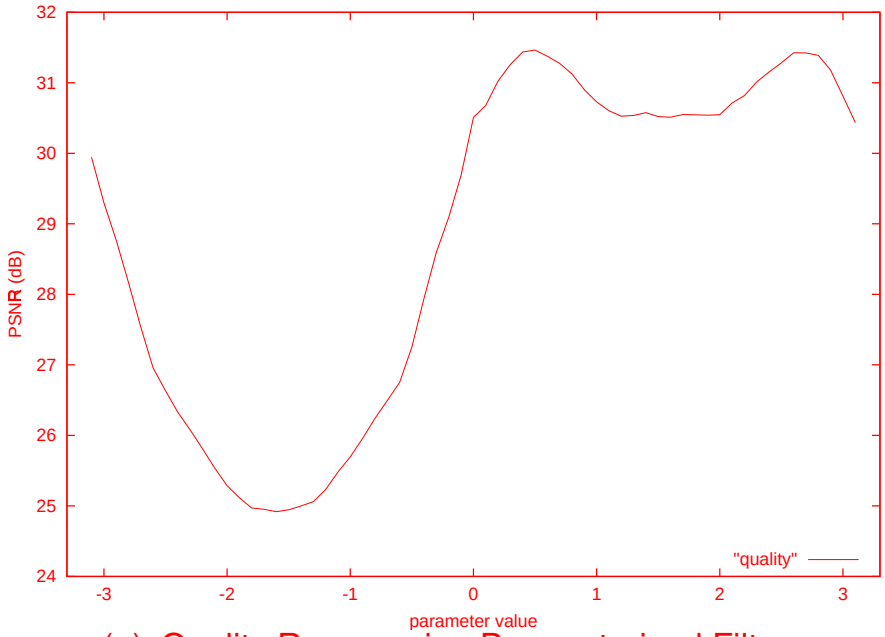
mit $c_k \in R$, und diversion Bedingungen an die Koeffizienten c_k. Eine Parametrisierung wurde von Schneid und Pittner [29] angegeben. Seien N Parameter $-\pi \leq \alpha_i < \pi, 0 \leq i < N$, gegeben, so kann folgende Rekursion benutzt werden, um die Filterkoeffizienten $c_k^N, 0 \leq k < 2N+2$ zu bestimmen:

$$c_0^0 = \frac{1}{\sqrt{2}} \text{ and } c_1^0 = \frac{1}{\sqrt{2}}$$

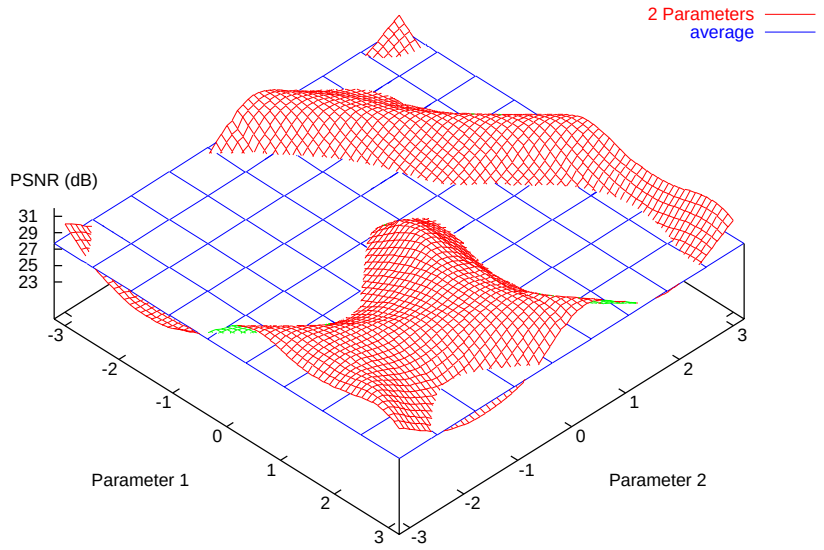
$$c_k^n = \frac{1}{2}\left((c_{k-2}^{n-1} + c_k^{n-1}) \cdot (1 + \cos\alpha_{n-1}) + (c_{2(n+1)-k-1}^{n-1} - c_{2(n+1)-k-3}^{n-1})(-1)^k \sin\alpha_{n-1}\right)$$

Wir setzen $c_k = 0$ für $k < 0$ und $k \geq 2N+2$.

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Parametrisierte Filter Beispiele 1



(e) Quality Range using Parameterised Filters



(f) Quality, using a 2-dimensional parameter space and average

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Parametrisierte Filter Beispiele 2

(g) Best Quality

(h) Worst Quality

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Parametrisierte Filter Sicherheit

Durch die parametrisierte Darstellung der Wavelet-Filterkoeffizienten, ist es möglich, eine symbolische inverse Wavelettransformation zu berechnen (das verschlüsselte Bild wird mit der inversen Transformation in den Bildbereich rücktransformiert) [11]. Als Ergebnis stehen an jeder Pixel Position ein symbolischer Ausdruck in den Paramtern der Filterkoeffizienten.

- Ciphertext-only Attacke: eine Funktion mit hohen Korrelation mit den Pixel Werten muss optimiert (minimiert) werden, z.B. Summe der Pixel Differenzen oder die Varianz. Ob das geht, hängt von der Art der verwendeten Parametrisierung ab.

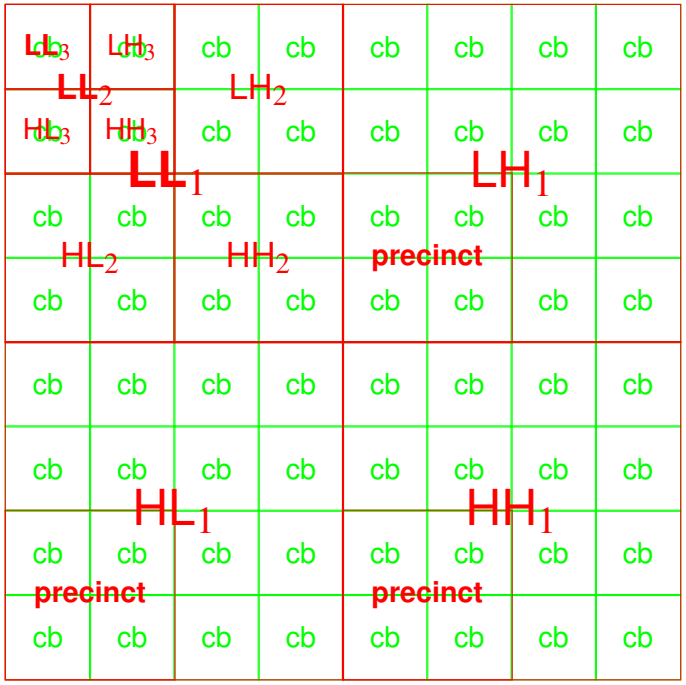
- Known Plaintext Attacke: durch Vorschaubild in geringer Qualität können einzelne Pixel Werte geschätzt werden (z.B. in grösseren homogenen Gebieten) oder es kann der durchschnittliche grauwert verwendet werden (nur 1 Paramter berechnebar). Die Art der Parametrisierung bestimmt die Anzahl der notwendigen Pixel für die Attacke.

Partielle Bildverschlüsselung durch geheime Wavelet Domains: Parametrisierte Filter - Angriffe

Ciphertext-only, Variance, 1 Par.

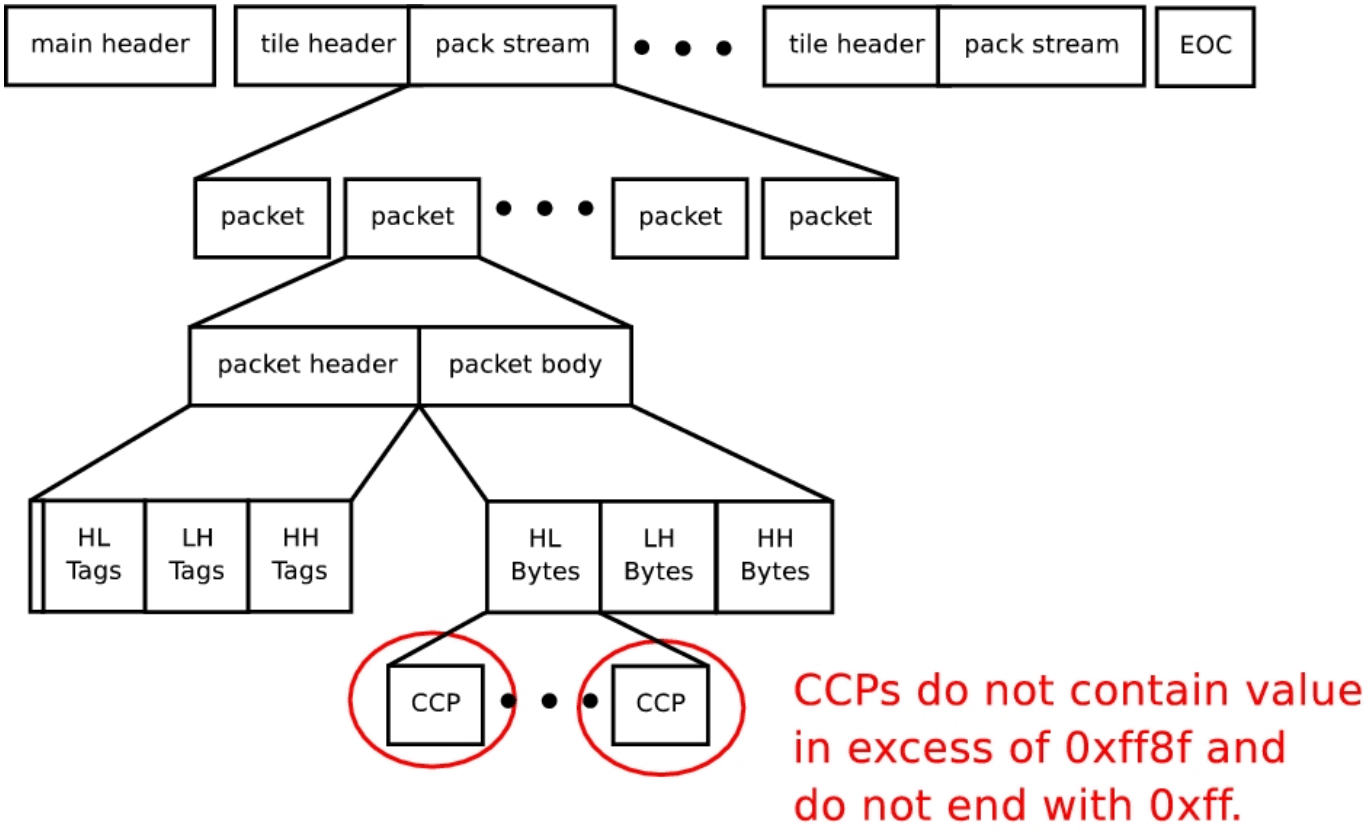
Known-Plaintext, Avg. Illumination, 1 Par.

Bitstreambasierte JPEG200 Verschlüsselung: Compliance I



Datenlayout des JPEG2000 Transformationsbereichs mit codeblocks (cb) und grösseren Strukturen.

Bitstreambasierte JPEG200 Verschlüsselung: Compliance II



Die Codestream Syntax erlaubt keine 2 byte Folge grösser als `0xff8f` und so eine Folge darf auch nicht mit `0xff` terminieren.

Bitstreambasierte JPEG200 Verschlüsselung: Compliance III

For all CCPs:

1. Encrypt the CCP.

2. Check if it contains a two byte sequence in excess of 0xff8f.
   If yes goto 1 and reencrypt the encrypted CCP.

3. Check if it ends with 0xff.
   If yes goto 1 and reencrypt the encrypted CCP.

4. Output the code-stream compliant encrypted CCP.

Das kann trivialerweise dekodiert werden (solange dekodieren, bis compliance wieder erhalten ist, denn der originale Datenstrom war auch compliant), jedoch muss bei den Kodierungs Settings darauf geachtet werden, dass die CCP nicht zu lange sind (sonst werden zuviele Iterationen notwendig und das Verfahren wird ineffizient) [34].

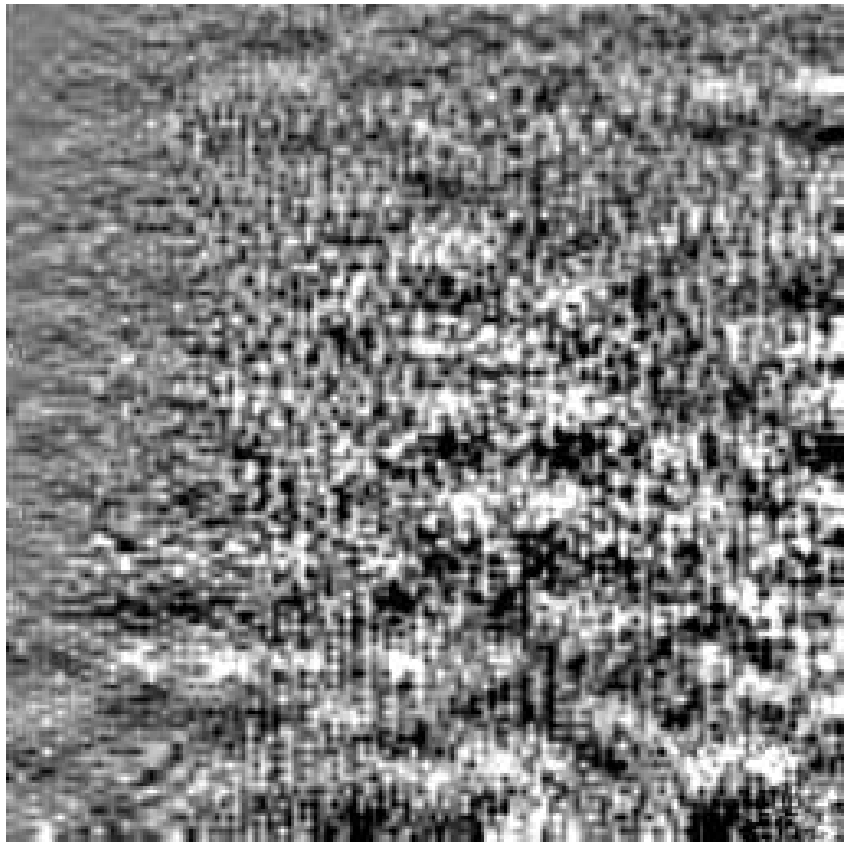
Bitstreambasierte Partielle Bildverschlüsselung: Wavelets J2K

Durch die Embeddedness des J2K Bitstreams kann eine partielle Verschlüsselung des Bitstreams sehr einfach durch Verschlüsselung der Anfangsteile des Bitstreams erreicht werden. Dadurch wird allerdings die format compliance zerstört und eine Kontrolle der verbleibenden Qualität ist nicht mehr durch einen gewöhnlichen J2K Dekoder möglich.

Daher wird in [14, 20] eine partielle Verschlüsselung der Packet Daten vorgeschlagen die sämtliche Headerstrukturen intakt lä[ßt. Die Realisierung in Norcen et al. [20] stützt sich auf JJ2000 und es wird die Eignung der resolution und layer progressive Modi für partielle Verschlüsselung verglichen.

Als Attacke gegen die partielle J2K Verschlüsselung wird in [20] die Replacement Attacke diskutiert (in Wen et al. [42] auch error concealment attack genannt), bei der die verschlüsselten Bitstreamteile ersetzt bzw. ignoriert werden. Das kann durch error concealment Methoden (durch bestimmte Marker im Bitstream werden fehlerbehaftete Teile bei der Dekompression ignoriert) von J2K einfach realisiert werden.

Bitstreambasierte Partielle Bildverschlüsselung: Wavelets J2K Beispiele I

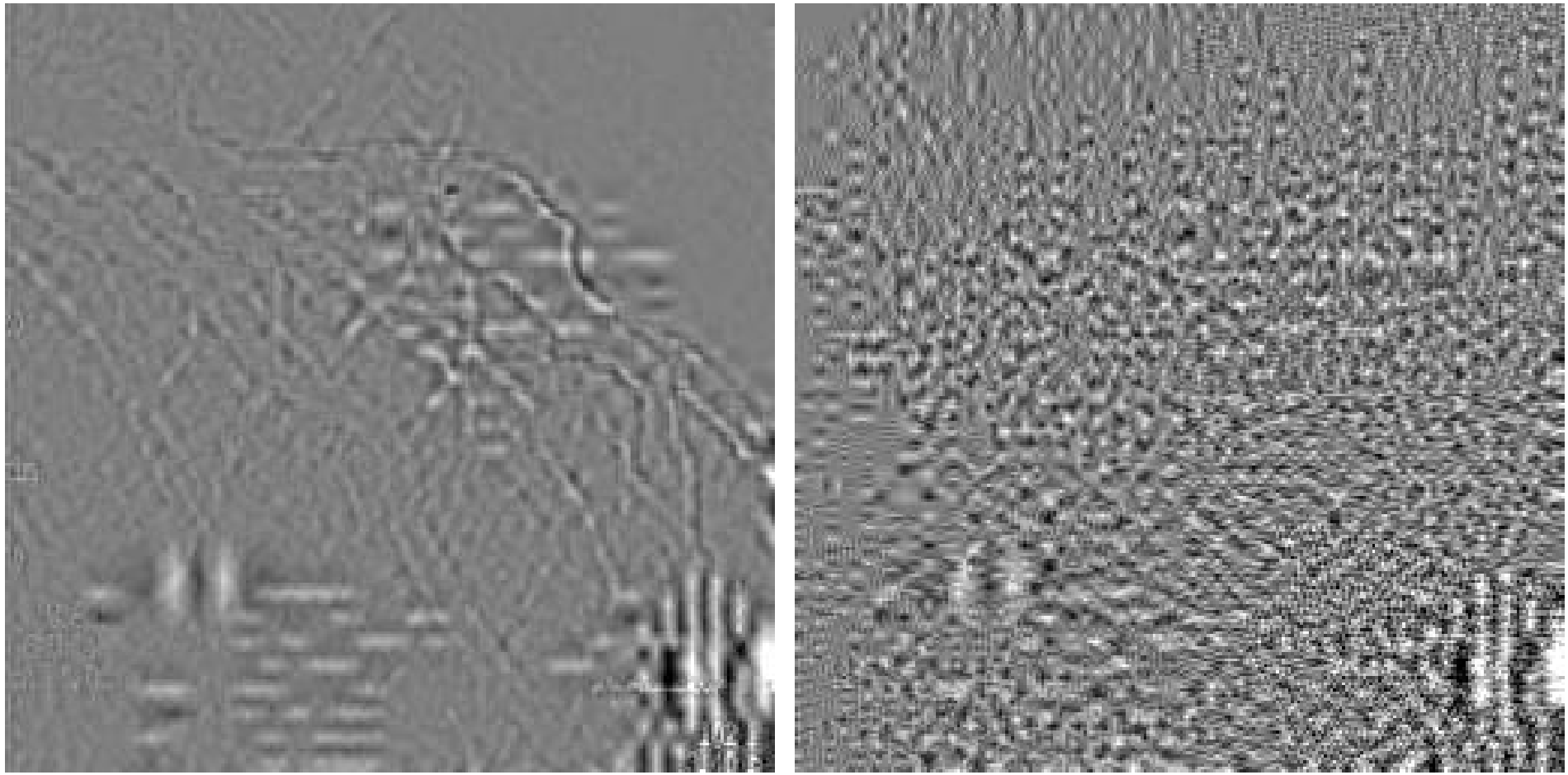


(i) layer progressive, 8.79 dB

(j) resolution progressive, 7.45 dB

Abbildung 8: Comparison of selective encryption (visual quality of reconstructed Angio2 where 1% of the bitstream data have been encrypted) using resolution or layer progressive encoding.

Bitstreambasierte Partielle Bildverschlüsselung: Wavelets J2K Beispiele II



(a) 1% encrypted, 10.51 dB

(b) 20% encrypted, 9.90 dB

Abbildung 9: Visual quality of reconstructed Angio2 after replacement attack using resolution encoding.

Bitstreambasierte Partielle Bildverschlüsselung: Wavelets SPIHT

Der Set Partitioning in Hierarchical Trees (SPIHT) Algorithmus ist eine fortgeschrittene Variante eines Zerotree Kompressionsverfahrens. Während der Kompression werden drei Listen verwaltet: nicht signifikante Pixelmengen (LIS), nicht signifikante (isolierte) Pixel (LIP) und signifikante Pixel (LSP). Diese Listen beschreiben die Zertotree Strukturen. Bei der Erzeugung des Bitstreams werden sign Bits, refinement Bits und significance Bits von Pixeln und Pixelmengen erzeugt.

Der Dekoder muß die zu dekodierenden Bits im korrekten Kontext (also als korrekten Typ) interpretieren um korrekt arbeiten zu können. Insbesondere inkorrekte significance Bits führen dazu, daß nachfolgende Bits falsch interpretiert werden, im Gegensatz zu falschen sign und refinement Bits. Hier setzt das Verfahren in [4] an und verschlüsselt die significance Bits der zwei tiefsten Subbands. Dadurch geht die notwendige Synchronisation zwischen Enkoder und Dekoder verloren und die nachfolgenden Bits können nicht mehr korrekt interpretiert werden. Hohe Sicherheit bei geringem Aufwand, wenn darauf geachtet wird kann format compliance erhalten werden.

Partielle Bildverschlüsselung mit Quadtrees I

Quadtree Bildkompression ist ein nicht-standardkonformes Verfahren, das schnell ist und im low-Bitratebereich bessere Ergebnisse als JPEG liefert. Kann sowohl als Bottom-up als auch als Top-down Verfahren realisiert werden.

1. Lossless Mode: Knoten im Baum werden zu Leaves wenn sie einheitlichen Grauwert haben und damit nicht weiter unterteilt werden müssen (Top-down approach). Gespeichert wird die Baumstruktur und die Grauwerte an den Leaves (mit identischer Anzahl von Bits für alle Leaves).

2. Lossy Mode: Anstelle des Tests auf einheitlichen Grauwert werden die potentiellen Leaves auf Ähnlichkeit überprüft (z.B. Pixelvarianz, ähnlicher durchschnittlicher Grauwert, ...). Die Anzahl der Bits zur Leave-value Repräsentierung ist variabel (also verlustbehaftet, hängt ab von der Größe des Leaves, der Varianz der Leave Werte auf dieser Quadtree Ebene , ...)

Im Lossless Fall ist für 8bpp Bilder 14% eine obere Schranke für den Anteil des Quadtrees an der gesamten Datenmenge, typische Werte im lossy Fall sind 15 – 25 % für einen großen Bitratenbereich.

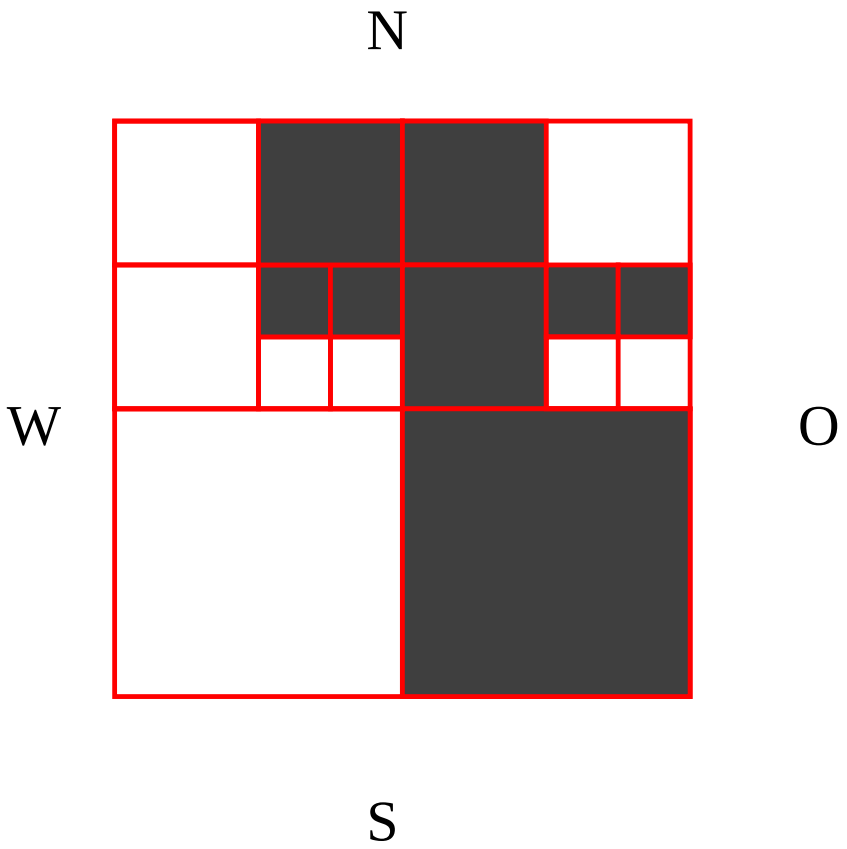
Partielle Bildverschlüsselung mit Quadtrees II

Cheng et al. [3, 4] schlagen vor, die Quadtreestruktur zu verschlüsseln und die Leavewerte unverschlüsselt zu lassen. Wie im Fall der Wavelet Packets ist auch hier eine brute force Attacke wegen der hohen Zahl der möglichen Quadtrees aussichtslos. Bleibt wieder der Ansatz, die Quadtreestruktur aus den nicht verschlüsselten Daten zu rekonstruieren. In diesem Zusammenhang werden zwei Varianten zur Organisation der Leave Werte diskutiert:

1. Leaf Ordering I: ist eine Depth First Anordnung beginnend mit dem NW Quadranten, immer gegen den Uhrzeigersinn.

2. Leaf Ordering II: zeilenorientierter Scan aller Leaves einer Quadtree Ebene, beginnend mit den kleinsten Leaves.

Partielle Bildverschlüsselung mit Quadtrees III



Leaf Ordering I: 0010011011110010

Leaf Ordering II: 1111000001100101

Abbildung 10: Example for leaf ordering I and II.

Partielle Bildverschlüsselung mit Quadtrees IV

Es stellt sich heraus, daß es einen Angriff gegen leaf ordering I gibt, wogegen gegen leaf ordering II keiner bekannt ist. Grundlage dieses Angriffes ist, daß in leaf ordering I die Leave values von benachbarten leaves in Bitstream ebenfalls benachbart sind. Aus dieser Tatsache ergibt sich, daß Runs (d.h. eine Folge von identischen leaf values) Informationen über Teilbäume geben, was die gesamte Anzahl der in Frage kommenden Quadtrees stark reduziert.

Ausgenutzt wird die Tatsache daß vier identische leaf values nicht zu Leaves auf der gleichen Quadtreeebene gehören können weil sonst kein Aufteilen stattgefunden hätte. Zusätzlich können homogene Regionen rekostruiert werden.

Partielle Verschlüsselung: Videos

Alle bisher beschrieben Verfahren lassen sich natürlicherweise auf I-Frames (und im Prinzip auch auf P und B-Frames) von Videos anwenden.

Zusätzlich werden aber videospezifische Eigenschaften und Strukturen zur Verschlüsselung verwendet (alle diese Verfahren können (durchaus sinnvoll) kombiniert werden):

- GOP Struktur: hier wird die unterschiedliche Wichtigkeit von I, P und B-Frames bezüglich des Informationsgehaltes ausgenutzt um eine partielle Verschlüsselung durchzuführen. Prägt den Begriff der “Selektiven Verschlüsselung”.

- Motion Vektoren: Da MV beschreiben wie Predictions für P und B-Frames aus I-Frames (oder P-Frames) gebildet werden, sind MVs der für die Videowahrnehmung wichtigere Teil von P und B Frames (verglichen mit dem komprimierten Fehlerbild).

Partielle Verschlüsselung: Selektive Video Verschlüsselung I

Maples und Spanos [33] schlagen als erste die Beschränkung von MPEG Verschlüsselung auf Verschlüsselung von I-Frames vor. Da I-Frames auch die Grundlage der Prediction für die bewegungskompensierten Frametypen sind, sollte dadurch der MPEG Stream ausreichend geschützt sein.

Schon bald wird von Agi und Gong [1] gezeigt daß dieser Ansatz zu unsicher ist, da durch die in P und B-Frames eingebetteten I-Blöcke (im Fall zu schlechter Prediction) zu viel Bildinformation erhalten bleibt. Daher sollen auch diese I-Blöcke verschlüsselt werden (siehe z.B. in Wu et al. [43] die Kombination eines solchen Ansatzes mit Watermarking).

Partielle Verschlüsselung: Selektive Video Verschlüsselung II

Meyer und Gadegast [27] entwickeln 1993 - 1995 SECMPEG das die Idee der selektiven Videoverschlüsselung aufgreift und auf DES, RSA und CRC (für zusätzliche Integritätsprüfungen) basiert. Für SECMPEG wurde ein eigenes Fileformat entwickelt, das zwar auf MPEG basiert, aber nicht kompatibel ist, d.h. man braucht eigene Encoder/Decoder Einheiten.

SECMPEG stellt vier Sicherheitsniveaus zur Verfügung:

1. Header verschlüsseln
2. Header, DC und die ersten ACs von I-Frames verschlüsseln
3. Alle I-frame Blöcke verschlüsseln
4. Alles verschlüsseln

Durch die nicht-Verschlüsselung der Motionvektoren in allen diesen Ansätzen bleibt Bewegung (von Objekten oder der Kamera) immer erkennbar.

Partielle Video Verschlüsselung: Motion Vektoren I

Werden zusätzlich zu I-Frames nur MVs verschlüsselt bleibt das Problem der I-Blöcke in P- und B-Frames. MV Schutz ist also v.a. also eine Zusatzmaßnahme – zusätzlich entsteht durch 0 setzen aller MVs meist eine relativ gute Frameapproximation (error concealment Attacke). Wie können MVs geschützt werden:

- Zeng und Lei [44] schlagen vor der Kodierung eine Verschlüsselung der MV Sign Bits und eine Permutation der MVs vor. Dies führt zu Datenexpansion – allerdings moderat weil der Anteil der MVs und MV Sign Bits ohnehin nicht groß ist. Wird kombiniert mit Sign Bit Verschlüsselung und Koeffizientenpermutation.

- Wen et al. [42] verschlüsseln (wieder über Indexliste) VLC MV Codewords. Auch hier Datenexpansion weil in den normalen Codewordconcatinierungen sehr viel 0 (kurze Codes) sind. Wird mit anderen Verfahren kombiniert (DCT VLC Codeword Verschlüsselung und Permutation).

- Shi et al. [30] geben in Analogie zu ihrem Verfahren für I-Frames einen fixen Scan durch P- und B-Frame Daten an, der bei den Sign Bits der MVs beginnt und dann DCT Sign Bits steigender Frequenz umfaßt. So werden pro Macroblock 64 Bit verschlüsselt.

Partielle Video Verschlüsselung: Motion Vektoren II

Cheng und Li [4] setzen auch für die Verschlüsselung der Motion Vektoren auf Baumstrukturen (I-Frames werden in dieser Arbeit durch teilweises Verschlüsseln des SPIHT Streams und Quadtree Files geschützt).

Idee ist, das Motion Vektor Field als Quadtree darzustellen, die Leaf values sind dann keine Grauwerte sondern eben einzelne Vektoren. Die Quadtreestruktur ist zu verschlüsseln, die leaf values bleiben im Plaintext. Die Autoren räumen allerdings ein, daß insbesondere bei kleinen Videoformaten die wenigen MVs (1 pro Makroblock) zu zu kleinen Quadtrees führen, sodaß eine vollständige Verschlüsselung notwendig wird. Besonders wichtig ist hier die Verwendung von leaf ordering II weil im MV Field viele homogene Regionen auftreten.

Spezielle Verschlüsselungs Verfahren

In den bisher besprochenen Verfahren wird auf die besonderen Eigenschaften von visuellen Daten durch die Verwendung von schnellen Ciphern (Permutationen) oder die Beschränkung auf wesentliche Teile eines Datenstromes eingegangen. Im Folgenden werden zwei spezielle Verfahren besprochen, die sowohl für Stillbilder als auch für Videos verwendet werden können.

- Bildverschlüsselung durch Chaotische Abbildungen: eine Klasse von chaotischen Abbildungen mit starken Durchmischungseigenschaften wird für Verschlüsselung von visuellen Daten verwendet (“chaotic mixing”).

- Fehlerrobuste Verschlüsselung für wireless Channels: treten bei der Verwendung von klassischen Ciphers Fehler im Ciphertext auf, zerstört dies meist eine relativ große Menge Plaintext. Das ist für wireless Channels (und andere fehleranfällige Speicher- und Übertragungmethoden) nicht günstig.

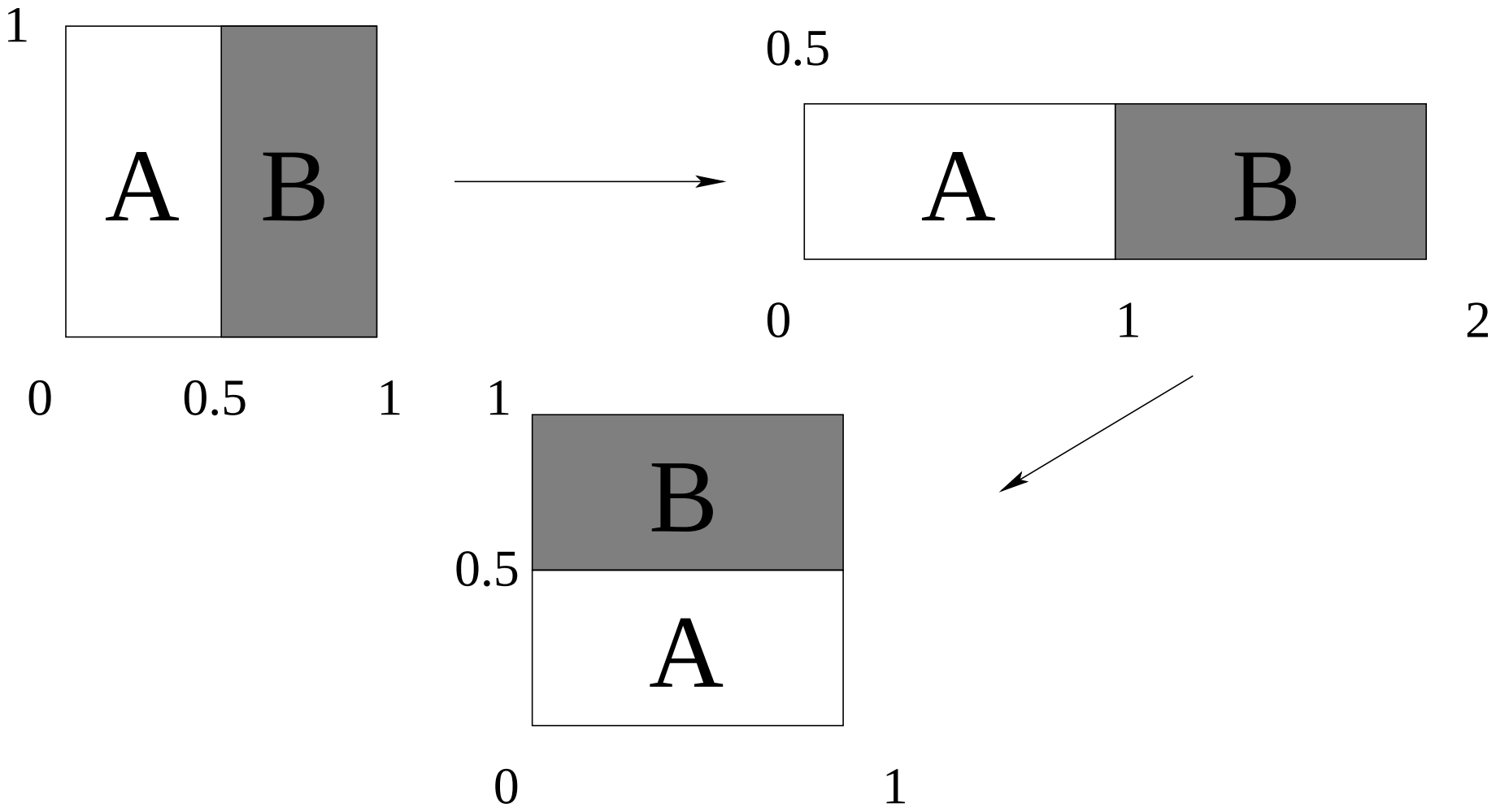
Spezielle Verfahren: Chaotic Mixing I

Grundlage dieser Klasse von Verschlüsselungsverfahren [28] ist die Tatsache, daß chaotische Abbildungen Eigenschaften besitzen, die denen von Verschlüsselungsverfahren sehr ähnlich sind, mit dem Hauptunterschied, daß es sich um stetige Prozesse handelt. Eine Diskretisierung von chaotischen Abbildungen ist die Grundlage solcher Verfahren. Das bekannteste Beispiel einer solchen Abbildung ist die "Baker Map" B auf $[0,1]^2$, die wie folgt definiert ist:

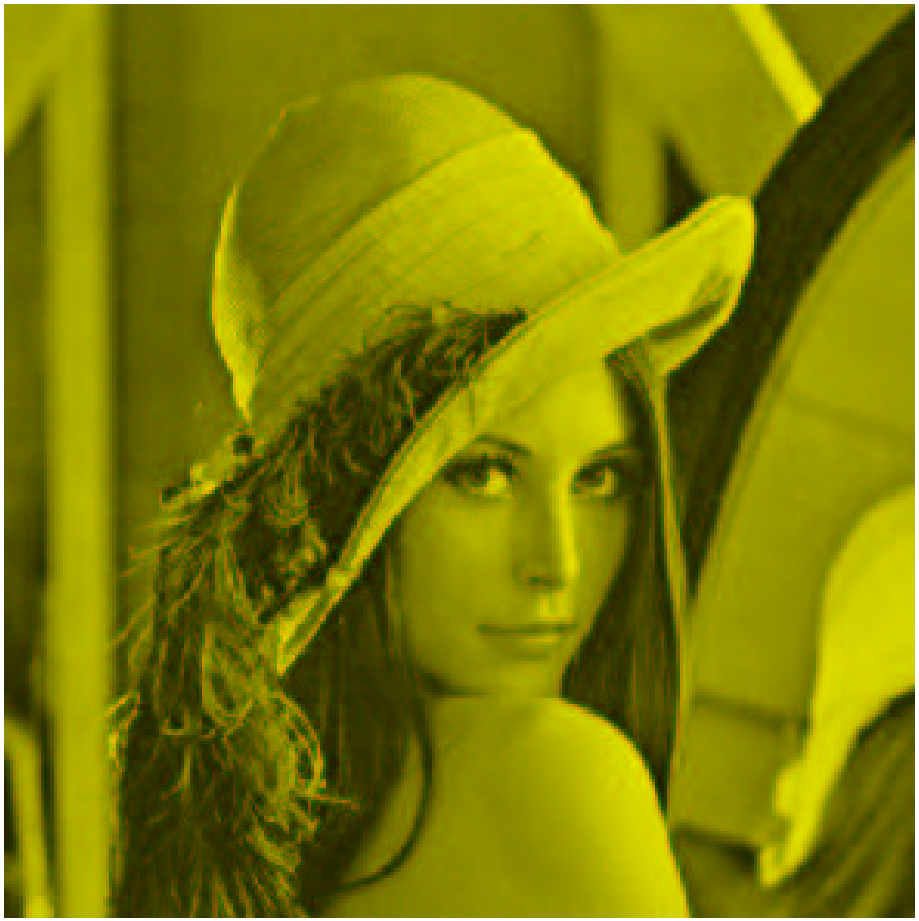
$B(x,y) = (2x, y/2)$ für $0 \leq x < 1/2$ und $B(x,y) = (2x-1, y/2+1/2)$ für $1/2 \leq x \leq 1$.

Die linke vertikale Hälfte $[0,1/2) \times [0,1)$ wird horizontal gestreckt und vertikal kontrahiert und auf den Bereich $[0,1) \times [0,1/2)$ abgebildet. Analog wird die rechte Hälfte $[1/2,1) \times [0,1)$ auf $[0,1) \times [1/2,1)$ abgebildet.

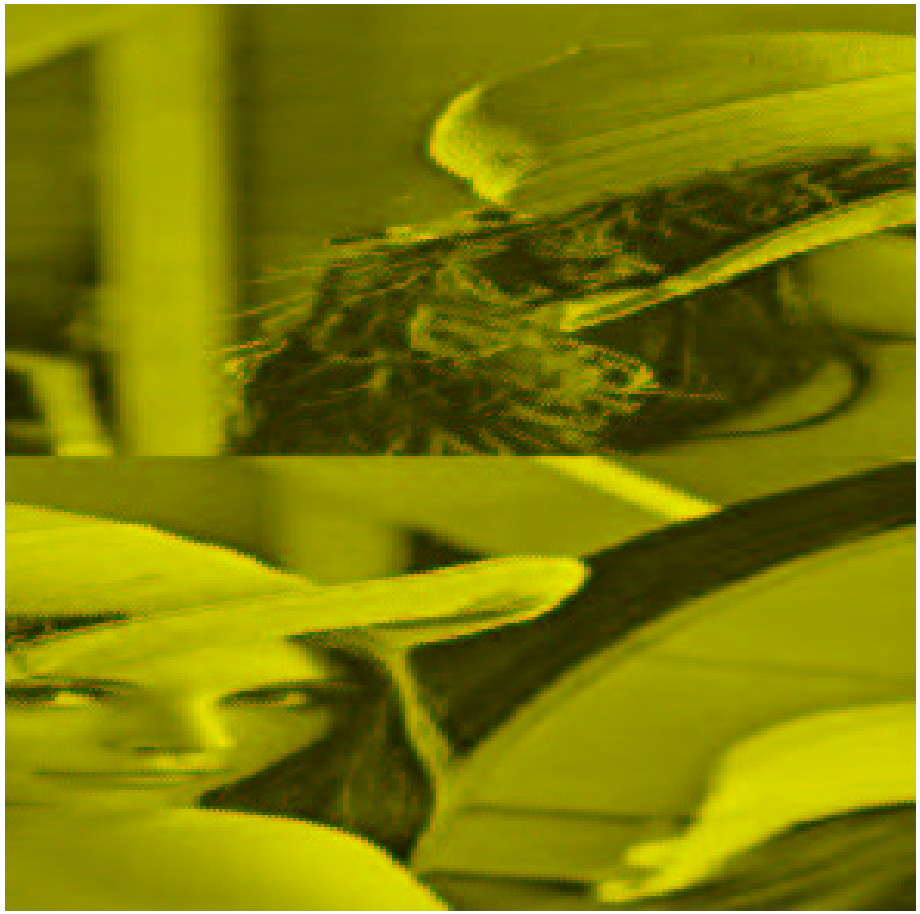
Spezielle Verfahren: Chaotic Mixing II



Spezielle Verfahren: Chaotic Mixing Beispiele 1



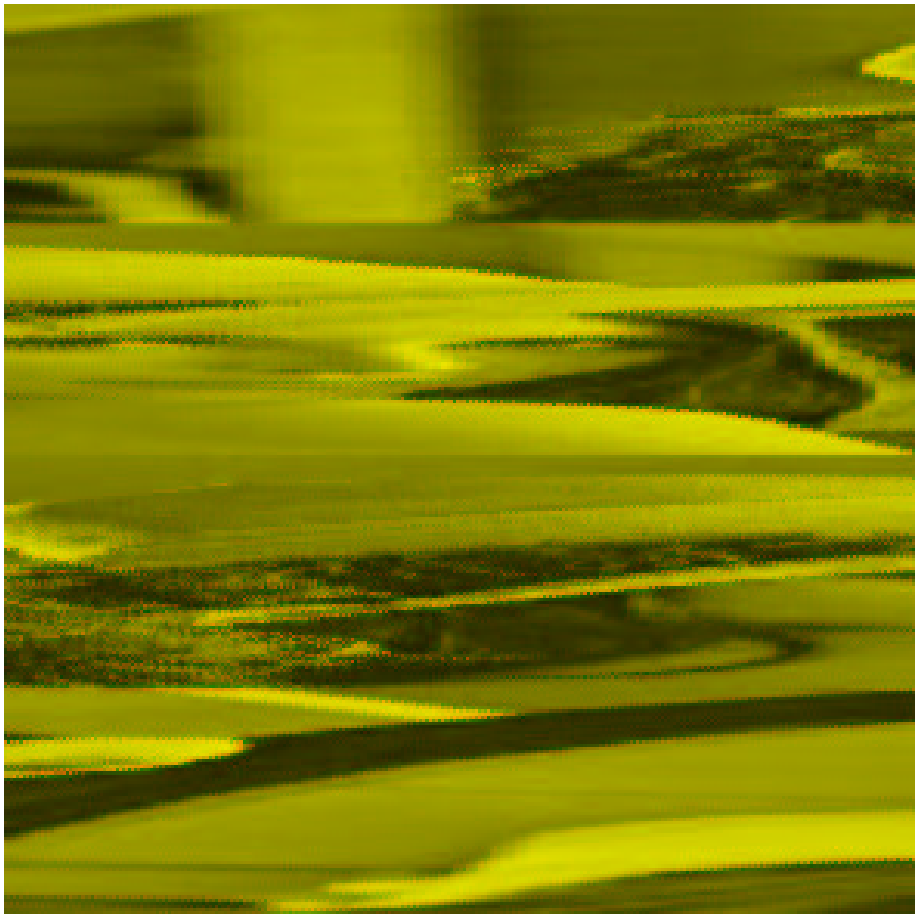
(a) Lena (gelb ;-)



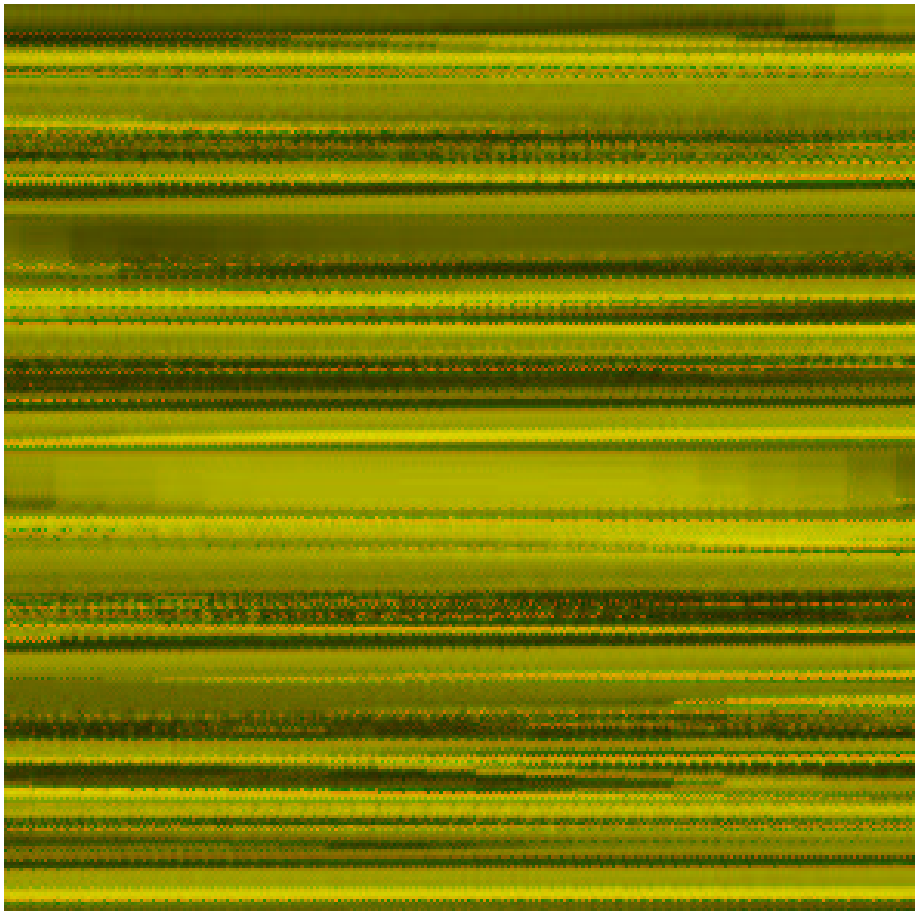
(b) 1 Iteration

Abbildung 11: Baker Map (1/2,1/2).

Spezielle Verfahren: Chaotic Mixing Beispiele 2



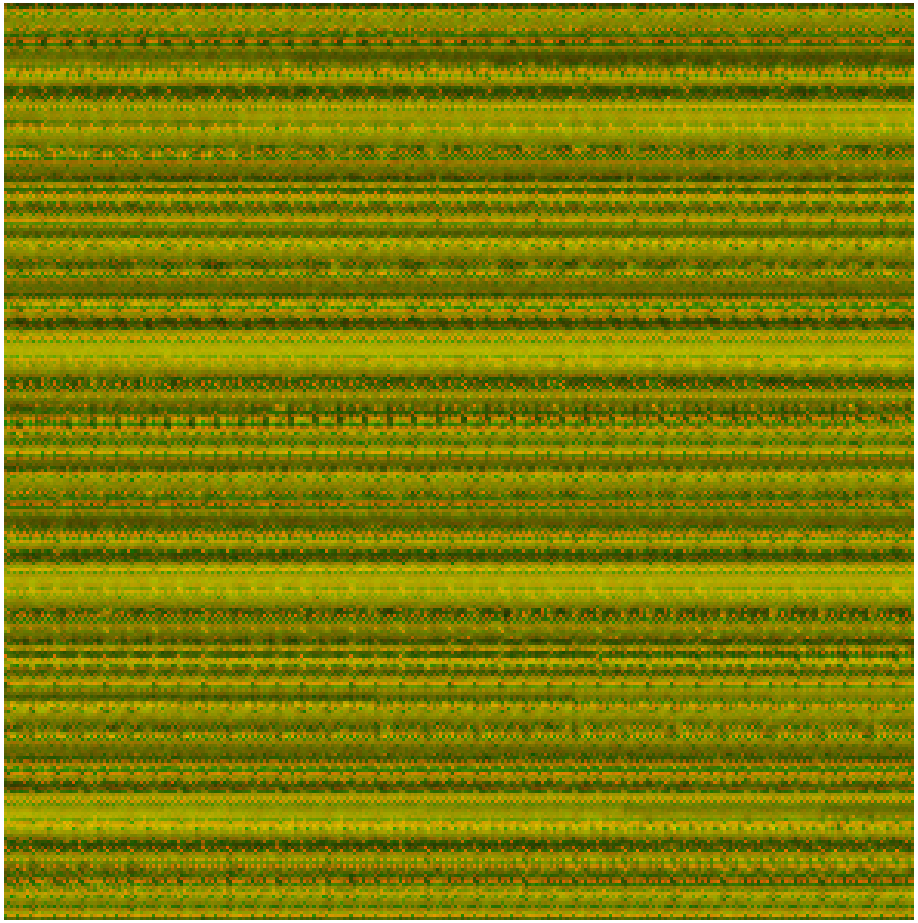
(a) 2 Iterationen



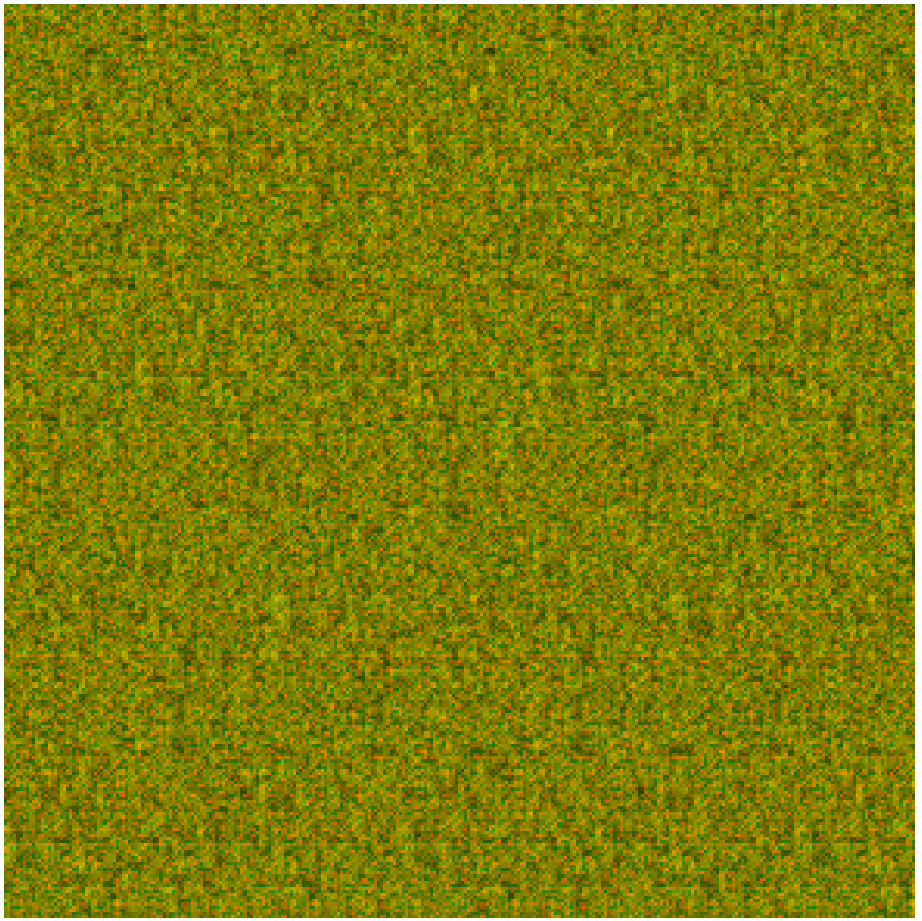
(b) 4 Iterationen

Abbildung 12: Baker Map (1/2,1/2).

Spezielle Verfahren: Chaotic Mixing Beispiele 3



(a) 6 Iterationen



(b) 10 Iterationen

Abbildung 13: Baker Map (1/2,1/2).

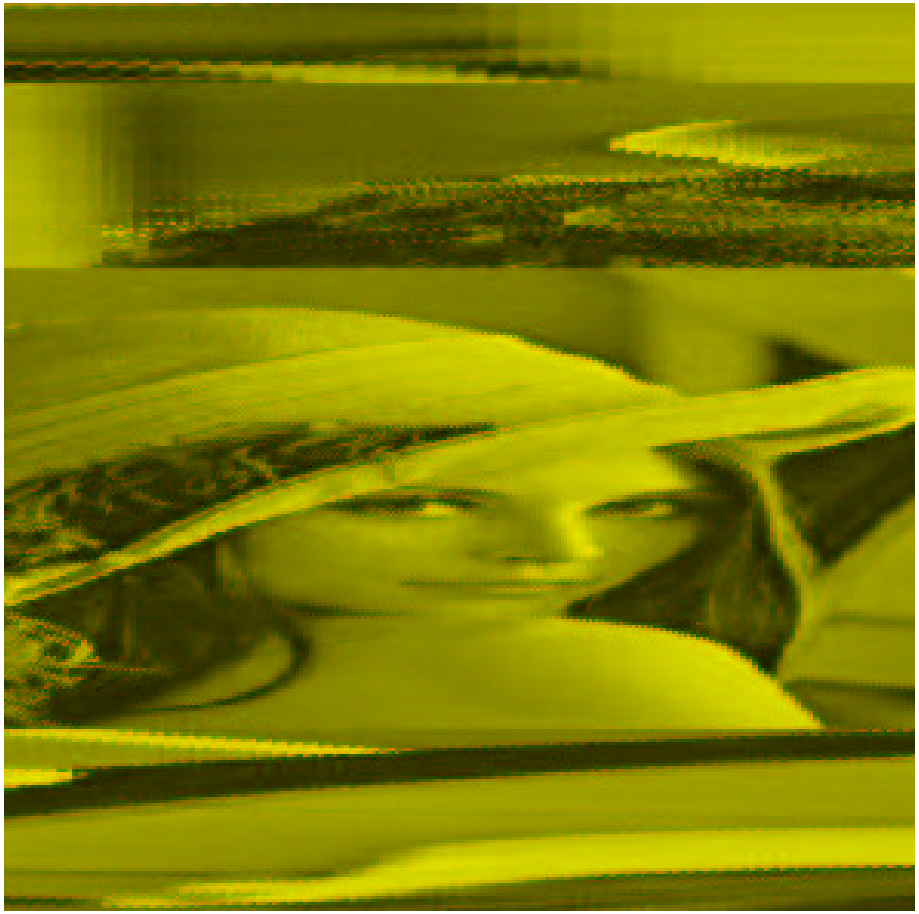
Spezielle Verfahren: Chaotic Mixing III

Die Baker Map kann wie folgt verallgemeinert werden: Anstelle der Betrachtung der 2 Hälften des Einheitsquadrats wird es in k vertikale Rechtecke $[F_{i-1}, F_i) \times [0,1)$ für $i = 1, \ldots, k$ und $F_i = p_1 + p_2 + \ldots + p_i$, $F_0 = 0$, sodaß $p_1 + \ldots p_k = 1$. Das linke untere Eck des i-ten Rechtecks ist F_i. Diese Verallgemeinerung streckt jedes Rechteck horizontal um den Faktor $1/p_i$ und kontrahiert es vertikal um den Faktor p_i. Abschließend werden die Rechtecke "übereinandergestapelt".

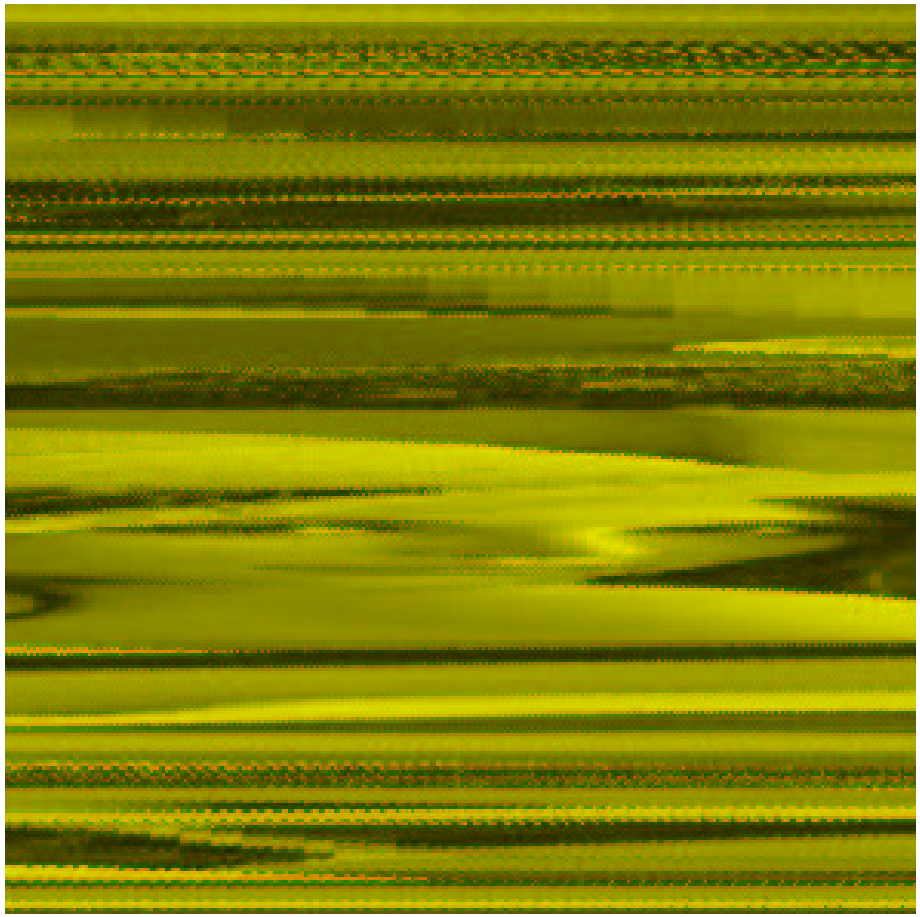
Um diese Abbildung auf ein Bild mit Pixelwerten anwenden zu können, benötigt man eine Diskretisierung, die eine bijektive Abbildung zwischen Pixeln sein muß. Dazu definiert man eine Folge von k ganzen Zahlen $n_1, \ldots, n_k$ sodaß jedes dieser n_i die Bildbreite N eines quadratischen Bildes teilt und $n_1 + \ldots + n_k = N$. $N_i = n_1 + \ldots + n_i$ und $N_0 = 0$. Das Pixel (r,s) mit $N_{i-1} \leq r < N_i$ und $0 \leq s < N$ wird abgebildet auf (wobei $q_i = N/n_i$):

$$(q_i(r - N_i) + s \pmod{q})_i, (s - s \pmod{q})_i)/q_i + N_i).$$

Spezielle Verfahren: Chaotic Mixing Beispiele 4



(a) 1 Iteration



(b) 2 Iterationen

Abbildung 14: Baker Map (1/10,1/5,1/2,1/5).

Spezielle Verfahren: Chaotic Mixing IV

Das bisher erhaltene Verfahren ist eine reine Pixelpermutation, definiert durch die diskretisierte chaotische Abildung wobei der Schüssel der Permutation (die Regel der Permutationstabellengenerierung) die Wahl der n_i ist. Um eine Durchmischung der Grauwerte zu erreichen wird die Abbildung eine drei-dimensionale erweitert. Ein Pixel (r,s) mit dem Grauwert g_{rs} wird auf $B(r,s)$ mit Grauwert $h(r,s,g_{rs})$ abgebildet, d.h. der neue Grauwert ist abhängig von der Pixelposition und dem ursprünglichen Grauwert. Um die Umkehrbarkeit zu erhalten, muß die Funktion h in der dritten Variable eine Bijektion sein, z.B. $h(r,s,g_{rs}) = g_{rs} + r * s \pmod{L}$ mit L die Anzahl der Grauwerte.

Dieses Verfahren ist nun eine Mischung aus Permutation und Substitution das nach wenigen Iterationen ein Bild mit gleichverteiltem Histogramm liefert. Um nun noch Diffusionseigenschaften zu erhalten (kleine Änderungen im Plaintext und Key sollen zu möglichst großen Änderungen im Ciphertext führen), wird ein einfacher nicht-linearer Feedback Shiftregistergenerator Spalte für Spalte angewendet ($g^*_{rs} = g_{rs} + G(g^*_{rs-1}) \pmod{L}$ mit beliebigem Seed). Der Schlüssel des gesamten Systems besteht aus den Parametern der chaotischen Abbildung, der Iterationsanzahl, der Parameter der Grauwerttransformation h und der Parameter des Feedback Registers.

Spezielle Verfahren: Chaotic Mixing V

Zusammenfassend hat das Verfahren folgende Eigenschaften:

- Arbeitet im Bildbereich ohne involvierte Kompression (siehe Szenario D). Allerdings könnte das Verfahren auch auf (quantisierte) Koeffizienten anstelle von Permutationen angewendet werden.
- Hoher Speicheraufwand da Transformationen auf gesamtem Frame operieren. Schlecht für Hardwareimlementierung.
- Hohe Geschwindigkeit durch Einfachheit der Operationen.
- Sicherheit von chaosbasierter Kryptographie ist umstritten, aber jedenfalls besser als von reinen Permutationen oder Substitutionen.
- Bei kleinen Bildern ist der Keyspace klein.

Spezielle Verfahren: Fehlerrobuste Verschlüsselung I

Ausgangspunkt für die Entwicklung dieser Technik ist die Tatsache, daß bei Fehlern im Ciphertext im Fall der Verwendung von klassischen Ciphers eine relativ große Menge Plaintext zerstört wird. Die Ursache sind die für die Sicherheit von Ciphern wichtigen Eigenschaften.

- Ähnliche Nachrichten führen zu völlig unterschiedlichen Ciphertexten (wichtig gegen known Plaintext Attacken !)
- Avalanche effect: die Änderung eines Input Bits führt im Schnitt zur Änderung von 50% der Outoutbits.
- Vollständigkeit: jedes Ciphertext Bit hängt von allen Plaintext Bits ab.

Bei fehleranfällingen Netzwerken (z.B. wireless) ist der massive Einsatz von FEC (forward error correction) durch erheblichen overhead zu bezahlen, d.h. nicht wünschenswert.

Spezielle Verfahren: Fehlerrobuste Verschlüsselung II

Tosun und Feng [37] führen eine systematische Untersuchung durch, welche Verschlüsselungsverfahren E diese für wireless Umgebungen ungünstigen Eigenschaften nicht haben. Erwünscht wäre die Eigenschaft, daß wenn E(x) und E(y) sich durch c Bits unterscheiden, x und y sich nur durch d $\leq$ c Bits unterscheiden.

Eine Klasse von Verschlüsselungsverfahren die diese Eigenschaft besitzt ist diejenige, die den Unterschied zwischen zwei Ciphertexten beibehält bezüglich des Unterschieds zwischen den entsprechenden Plaintexten. Formal wird das über die Hamming Distanz $d(x,y)$ zwischen zwei binären Strings ausgedrückt: ein Verschlüsselungsverfahren wird als "error preserving" bezeichnet, wenn $d(x,y) = d(E(x),E(y))$. Im Folgenden werden Verschlüsselungsverfahren mit dieser erwünschten Eigenschaft charakterisiert.

Spezielle Verfahren: Fehlerrobuste Verschlüsselung III

Es wird gezeigt, daß alle error preserving Verfahren durch Bitpermutation und Bitkomplementierung erzeugt werden können. Dies illustriert sehr deutlich, daß diese Verfahren anfällig gegen die known Plaintext Attacke sind (ist klar, da sie die Eigenschaften die diese verhindern eben nicht besitzen, siehe oben).

Ähnlich wie beim VEA II wird über die statistischen Eigenschaften von MPEG Strömen argumentiert (*stationary und memoryless*) daß ciphertext only Attacken nicht durchführbar sind. Gegen die known Plaintext Attacke wird wird ein Parameterwechsel auf Framebasis vorgeschlagen.

Diese Arbeit zeigt, daß es v.a. vom Standpunkt der Fehlerausbreitung Sinn macht, für Videoverschlüsselung Permutationen zu verwenden (neben der hohen Geschwindigkeit). Methoden aus lokalen Daten entsprechende Permutationstabellen zu generieren um Robustheit gegen known Plaintext Attacken bei kleinem Aufwand für Schlüsselmanagement zu erreichen wurden in [42] gezeigt (siehe vorher).

Spezielle Verfahren: 3D Graphik Daten I

- Ein Mesh M ist ein Tupel (V,K)
- V ist eine Menge von Vertex Positionen v_i mit $v_i := (x,y,z)$; $x,y,z \in \mathbb{R}$ (Geometrie Information)
- K ist eine Menge von Flächen des Mesh wobei eine Fläche $k_j := (e_l, e_m, e_n)$ definiert ist als ein Tripel von Kanten von M (Connectivity Information)
- e_k ist ein Tupel von Vertices von M. $e_k := (v_i, v_j); v_i, v_j \in V$
- $Nb(v_i)$ sind die (unverschlüsselten) direkten Nachbarn von v_i. $Nb(v_i) := \{v_j \mid (v_i, v_j) \in M\}$
- $INb(v_i)$ sind die (unverschlüsselten) indirekten Nachbarn von v_i. Indirekte Nachbarn sind mit v_i durch einen dazwischenliegenden Vertex verbunden.
  $INb(v_i) := \{v_j \mid \exists v_k \in M : (v_j, v_k) \in M \wedge (v_k, v_i) \in M \wedge v_k \notin Nb(v_i)\}$

Spezielle Verfahren: 3D Graphik Daten II

Während es in der Literatur viele Ansätze zum Watermarking solcher Daten gibt ("second line of defense"), gibt es wenig im Bereich Verschlüsselung ("first line of defense"):

- Ein Ansatz erlaubt es dem Viewer eines Clients nur eine geringe Auflösung zu laden. Der Client sendet den gewünschten Viewing Winkel zu Server, der dann ein Bild der hochaufgelösten Daten rendert und an den Client schickt. Somit hat der Client keinen Zugang zu den hochaufgelösten Daten. Eine geschützte Übertragung oder Szenarien wie transparente Verschlüsselung ist mit diesem Ansatz aber nicht realisierbar.

- Ein zweiter Ansatz wendet partielle Verschlüsselung auf eine zufällig ausgewählte Menge von Vertices und Connectivity Inforamtionen an. Problematisch ist hier die geringe Sicherheit wenn wenige Daten geschützt werden und ein sehr hoher Verschüsselungsgrad, wenn hohe Sicherheit notwendig ist. Dies liegt in der hohen Redundanz der Daten begründet.

Grundlegende Idee in [16, 15]: Verwenden einer hierarchischen Mesh-repräsentierung analog zu skalierbaren bzw. embedded Medien Datenformaten.

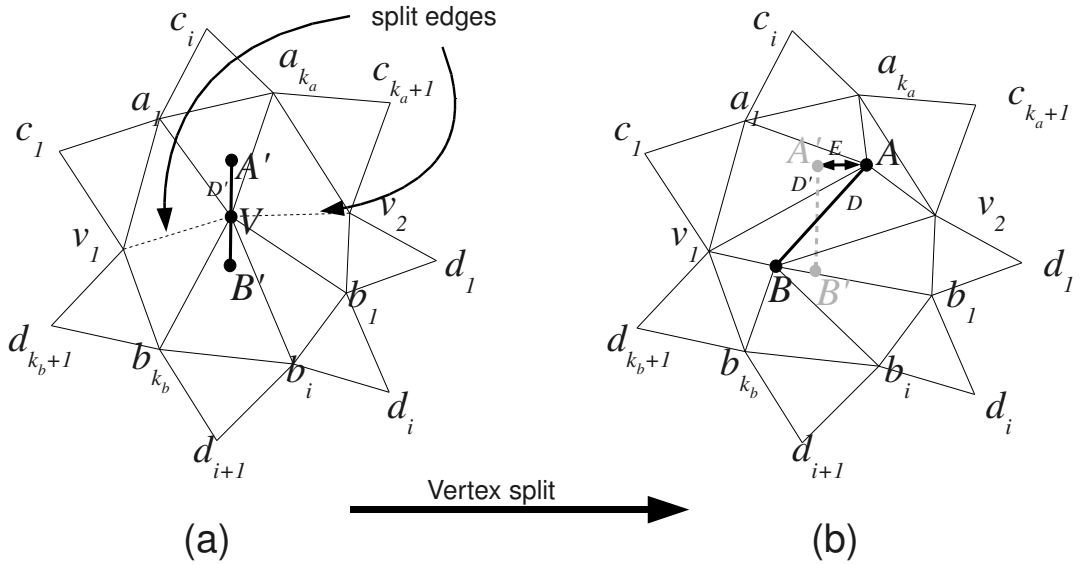
Spezielle Verfahren: 3D Graphik Daten III

1. Mesh Simplification: einfache Strategien um die Datenmenge z.B. für schnelles Rendering zu reduzieren.
   - Feature Removal: iteratives Entfernen von kleinen features
   - Vertex Clustering: mehrere Vertices werden durch einen einzelnen ersetzt

2. Progressive Meshes: weniger detaillierte Mesh Versionen können mit weniger Daten visualisiert werden und für die volle Auflösung werden die Daten der niederen Aufösung wiederverwendet.
   - Compressing progressive Meshes: aus den niederaufgelösten Daten wird eine Prediction der fehlenden Vertex Daten der höheren Auflösung berechnet und nur der Fehler wird gespeichert. Ziel ist bestmögliche Datenreduktion.
   - Non-compressing progressive Meshes: die zusätzlichen Daten der höheren Auflösung werden direkt gespeichert. Ziel ist geringer Rechenaufwand trotz des progressiven Formats.

Spezielle Verfahren: 3D Graphik Daten IV

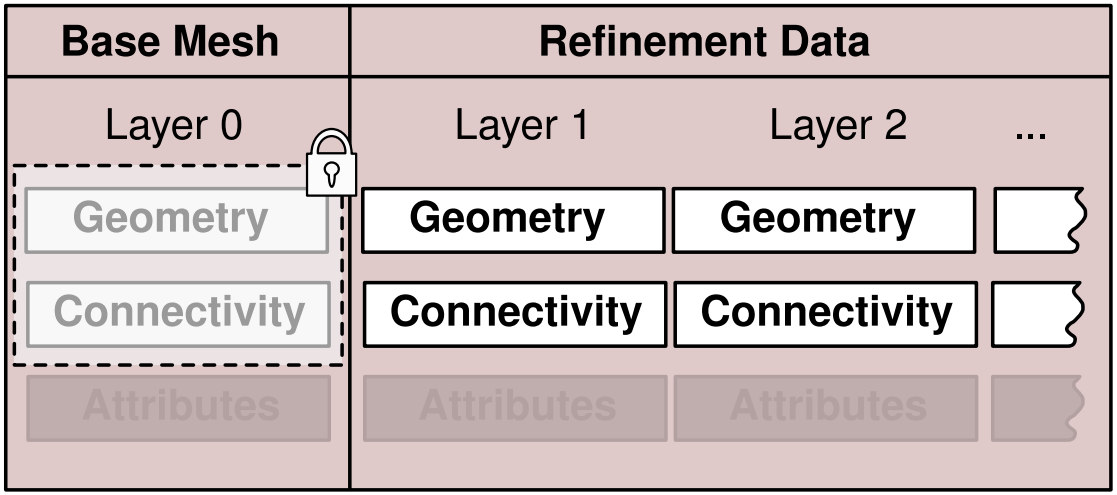
Wie werden die niedrigen Aufösungen bzw. die Predictions für die höheren Auflösungen erzeugt ?



1. Zwei Vertices A & B deren Entfernung den geringsten Impact auf das Mesh hat werden gewählt um sie zu entfernen (edge collapse).
2. Diese beiden werden zum Vertex V vereinigt, V wird als Split-Kandidat gespeichert. Die beiden Vertices (A', B') werde durch ihre Umgebung geschätzt (Vertex-split von V) (z.B. a_k, b_k, c_k, v_1 und v_2 werden verwendet um A' vorherzusagen).
3. Der Vektor $D = A - B$ ist der Distanzvektor, gespeichert wird nur der Fehlervektor $E = D - D'$ (mit $D = A' - B'$.

Spezielle Verfahren: 3D Graphik Daten V

Erste Verschlüsselungsstrategie betrachtet Non-compressing progressive Meshes, die Geometrieinformation wird verschlüsselt, wir beginnen vom Start des Files. Der nicht-verschlüsselte Teil des Files kann ohne Beeinträchtigung dekodiert werden.



Spezielle Verfahren: 3D Graphik Daten VI

Wie im Fall von visuellen Daten ist auch hier eine direkte Entschlüsselung nicht zielführend. Der einachste Angriff setzt die verschlüsselten Daten auf $(0.0, 0.0, 0.0)$ oder auf den Mittelwert aller nicht-verschlüsselten Vertices. Durch die Connectivity Information kann der Angreifer unverschlüsselte Vertices in der Nähe der verschlüsselten identifizieren und durch lineare Interpolation eine Vorhersage $P(v_i)$ berechnen.

$$Avg_1(v_i) = \sum_{v_k \in Nb(v_i)} \frac{v_k}{\#Nb(v_i)}, \quad Avg_2(v_i) = \sum_{v_k \in INb(v_i)} \frac{v_k}{\#INb(v_i)}$$

$$\text{if} \#Nb(v_i) > 1: \quad P(v_i) = Avg_1(v_i)$$

$$\text{otherwise}: \quad P(v_i) = \frac{2 * \#Nb(v_i) * Avg_1(v_i) + \#INb(v_i) * Avg_2(v_i)}{2 * \#Nb(v_i) + \#INb(v_i)}$$

Das Ergebnis ist analog wie die Interpolation eines verschlüsselten Pixels in einem Bild. Wenn mehr und mehr Vertices verschlüsselt werden, wird die Prediction immer schlechter.

Spezielle Verfahren: 3D Graphik Daten VII

Wie können Fehler gemessen werden ? Pixelweise (Vertex-weise) Unterschiede können nicht berechnet werden, da es nicht notwendigerweise zu jedem Vertex eine Entsprechung im zu vergleichenden Mesh gibt. Daher wird eine Variante der Hausdorff Distanz verwendet: das Verfahren verteilt zufällig eine Menge von von Punkten auf dem ersten Mesh M, für jeden dieser Punkte p wird die minimale Distanz zum zweiten Mesh M' berechnet. Das Maximum dieser Distanzen ist die Hausdorff Distanz.

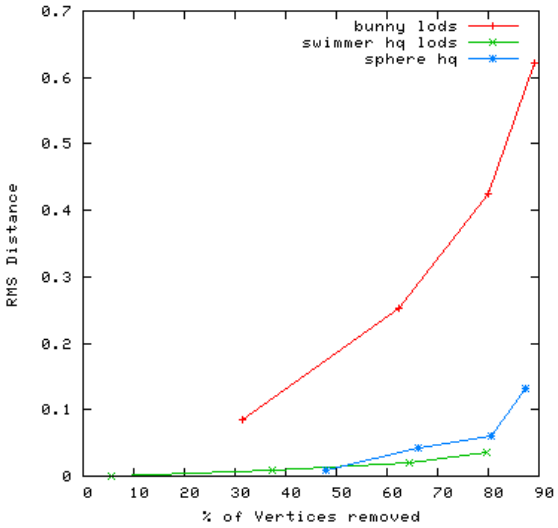
$$d(p,M') = \min_{p' \in M'} \|p - p'\|_2$$

$$d(M,M') = \max_{p \in M} d(p,M')$$

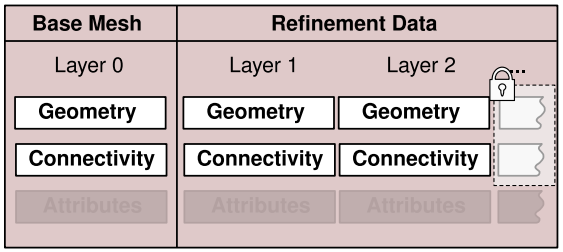
Da hier statistische Outlyer überbetont werden, wird stattdessen der Durchschnitt über die Punkte des Mesh M berechnet:

$$d_{rmse}(M,M') = \sqrt{\frac{1}{|M|} \sum_{p \in M} d(p,M')^2}$$

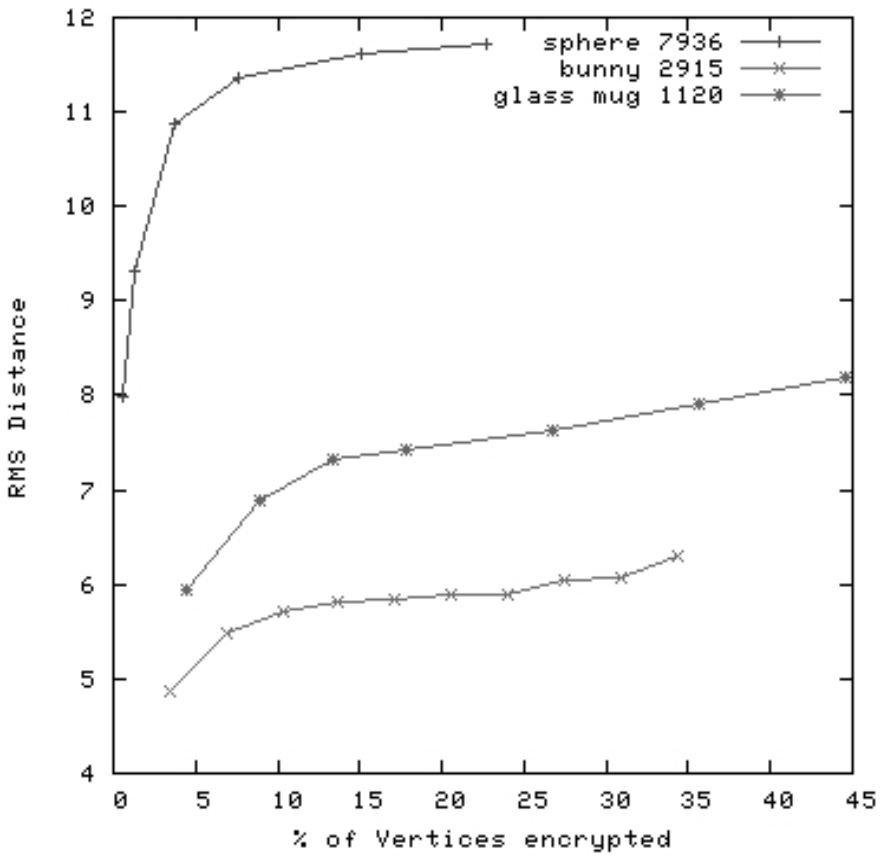
Spezielle Verfahren: 3D Graphik Daten VIII

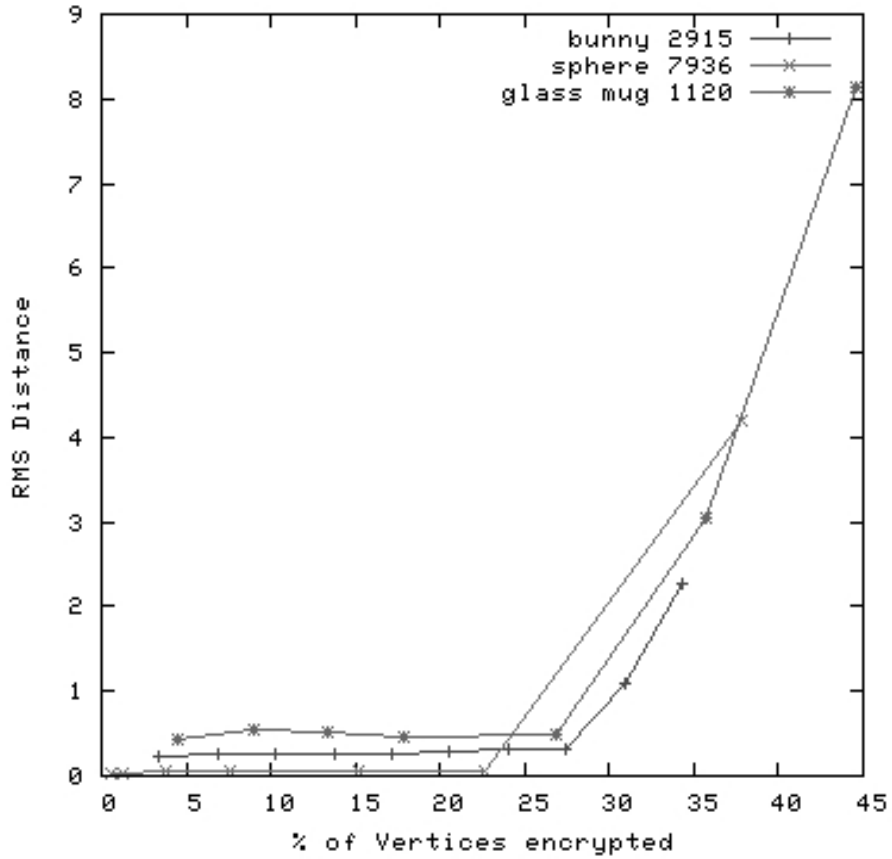


Das Fehlermass funktioniert demnach wie es soll. Je mehr Vertices weggelassen werden, desto grösser wird der Abstand. Das entspricht einer Verschlüsselung von mehr und mehr Refinement Daten (der einfachste Ansatz, wie klassische transparente Verschlüsselung von skalierbaren Medien).



Spezielle Verfahren: 3D Graphik Daten IX





RMS distance when encrypting (starting at the “top” of the file) and attacking a non-compressing mesh – the technique employing the average midpoint leads to significant errors, however, the attack using neighbourhood data provides low errors for up to 30% of data encrypted.

Spezielle Verfahren: 3D Graphik Daten X

Offensichtlich ist die Attacke mit der Durchschnittsbildung wesentlich effizienter, wie korrekt durch das Fehlermass vorhergesagt (die ersten 500 Vertices wurden verschlüsselt und entsprechend ersetzt).

Spezielle Verfahren: 3D Graphik Daten XI

Zweite Verschlüsselungsstrategie betrachtet Compressed Progressive Meshes (in zwei Varianten: base mesh Verschlüsselung und Verschlüsselung des ersten Refinement layer); hier werden die Positionen der Vertices v_i der höheren Qualität vorhergesagt und nur der Prediction-Fehler E_{v_i} gespeichert.

$$E_{v_i} = v_i - P(v_1, v_2, .., v_{i-1}, v_{i+1}, .., v_n)$$

Die Vertices die nach den verschlüsselten Daten dekodiert werden basieren auf falschen Vorhersagen und sind damit inkorrekt:

$$v'_i = P(v'_1, v'_2, .., v'_{i-1}, v'_{i+1}, .., v'_n) + E_{v_i}$$

$$v'_i = v_i - P(v_1, v_2, .., v_{i-1}, v_{i+1}, .., v_n) + P(v'_1, v'_2, .., v'_{i-1}, v'_{i+1}, .., v'_n)$$

Offenbar spielt also die Art des Prediktors eine wichtige Rolle bei der Datenrekonstruktion.

Spezielle Verfahren: 3D Graphik Daten XII

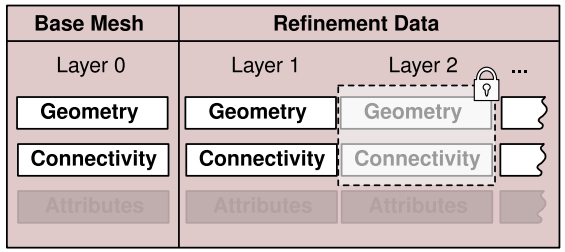
- Linear prediction: für $v_j \in Nb(v_i)$ ist $a_j = 1$, sonst 0:

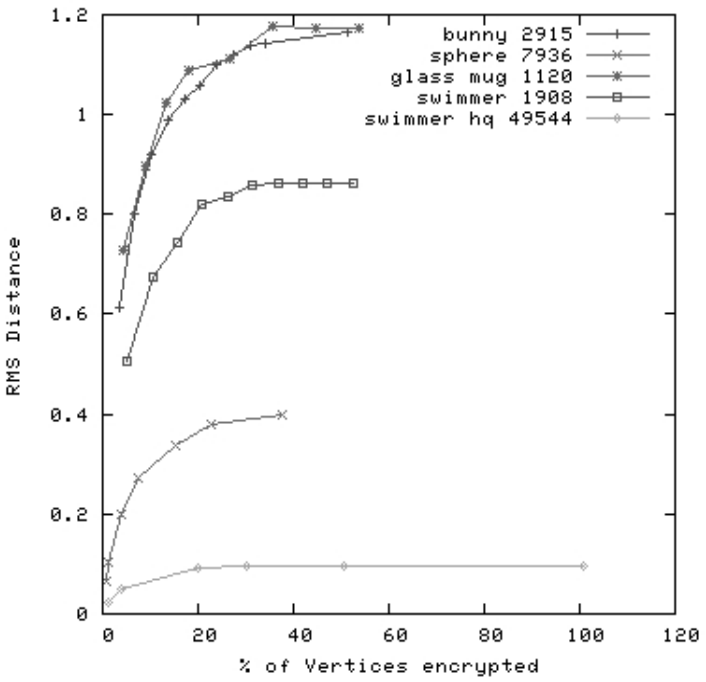
$$P_L(v_1,..,v_{i-1},v_{i+1},..,v_n) := \frac{a_1v_1 + .. + a_nv_n}{\#Nb(v_i)}$$

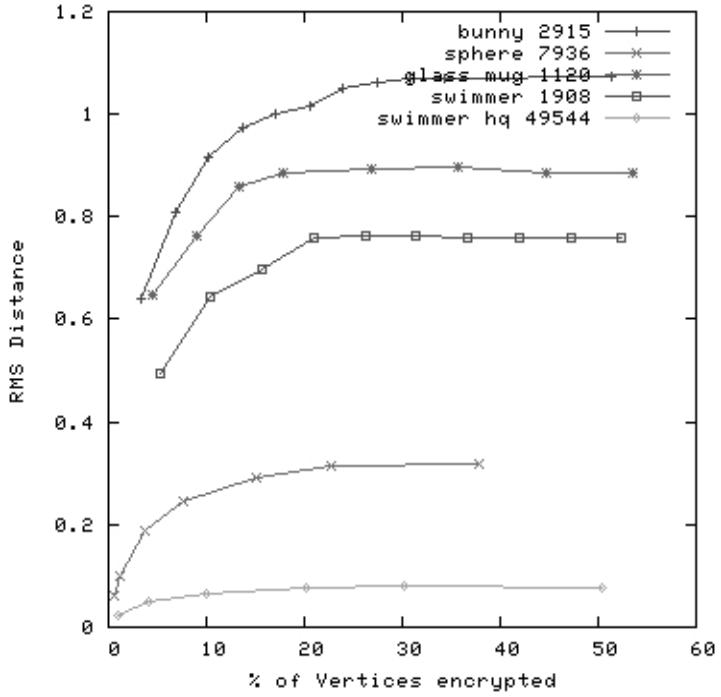
- Butterfly prediction: für $v_j \in Nb(v_i)$ is $a_j = 1$, sonst 0 und für $v_j \in INb(v_i)$ ist $b_j = 1$, sonst 0 ($\alpha = 1.15$ ist typisch):

$$P_L(v_1,..,v_{i-1},v_{i+1},..,v_n) := \alpha * \frac{a_1v_1 + .. + a_nv_n}{\#Nb(v_i)} + (1-\alpha) * \frac{b_1v_1 + .. + b_nv_n}{\#INb(v_i)}$$

Spezielle Verfahren: 3D Graphik Daten XIII



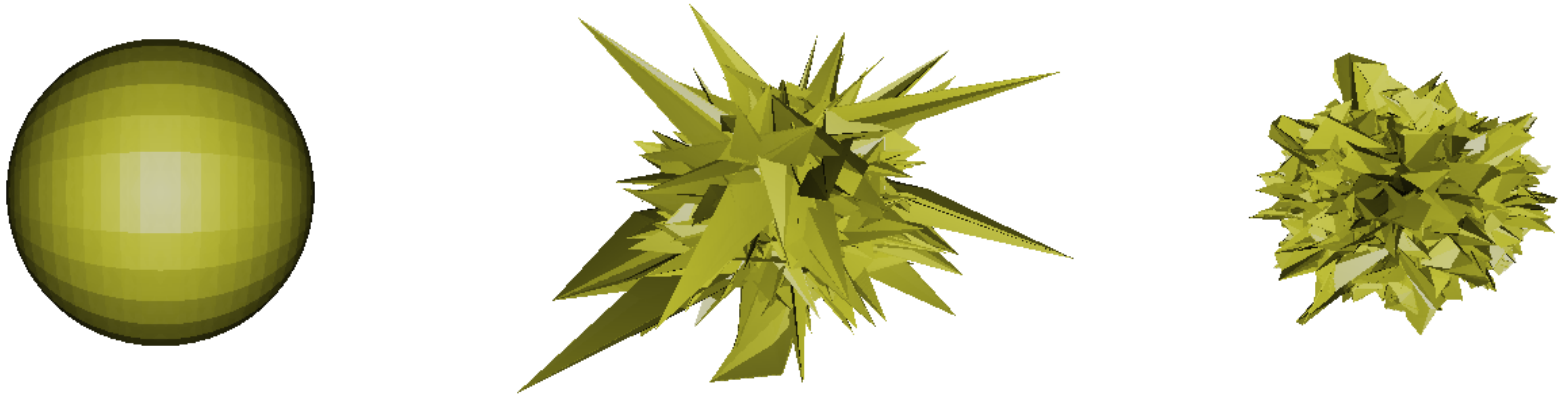




Der Unterschied zwischen den Prediktoren ist klar ersichtlich; offenbar ist diese Strategie nur für transparente Verschlüsselung einsetzbar.

Spezielle Verfahren: 3D Graphik Daten XIV

Für content confidentiality scheint base layer Verschlüsselung von compressing progressive meshes durchaus geeignet zu sein:



Der Unterschied zu Bild- und Videoinformation: Es ist nicht klar, was als “typische” Daten für das Base Mesh verwendet werden könnten. Auch eine Rekonstruktionsattacke (Mesh-Glattheit als Kriterium) scheint schwierig.

Beurteilung von Verschlüsselungsverfahren I

Um die Sicherheit von Verschlüsselungsverfahren beurteilen zu können, muss zuallererst Klarheit geschaffen werden, welche Bedeutung “Sicherheit” haben soll.

- MP security (message privacy): Wissen über den Ciphertext darf zu keinerlei Erkenntnisgewinn über den Plaintext führen. Hier dürfen keinerlei komprimierte Mediendaten verschlüsselt werden, da schon die Dateigrösse diverse Rückschlüsse zulässt.

- MR security (message recovery): hier ist das Ziel die Vermeidung der vollständigen Wiederherstellung des Plaintexts. Verschlüsselung des hinteren Endes eines Skalierbaren Bitstroms ist hier gleich MR-sicher wie die vollständige Verschlüsselung.

- MQ security (message quality): Ziel eines Angriffes ist hier eine gute *approximative* Rekonstruktion der Daten mit guter perzeptueller Qualität.

Der letzte Sicherheitsbegriff inkludiert einen Qualitätsbegriff, sodass dieser näher betrachtet werden muss.

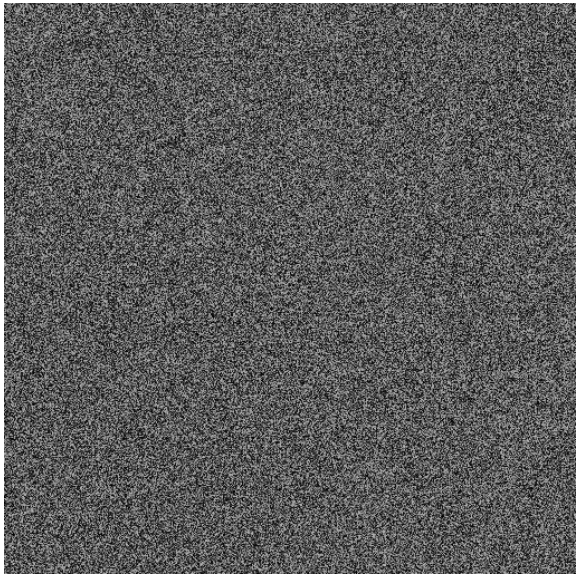
Beurteilung von Verschlüsselungsverfahren II

Qualitäts-Metriken wurden bisher typischerweise für mittel- bis hochqualitative Daten entwickelt, die wenigen Ausnahmen zeigen noch keine überzeugendes Verhalten:

- Mean opinion score (MOS): durchschnittliche Qualitätsbeurteilung durch menschliche Betrachter. Alle numerischen Masse sollte hohe Korrelation mit diesem Wert aufweisen.
- PSNR: pixelweiser quadratischer Unterschied; funktioniert schon für hohe Qualität schlecht, für tw. verschlüsselte Daten sehr fragwürdig.
- Structural similarity index (SSIM): kombiniert Helligkeit, Kontrast und Strukturinformationen in einen finalen Wert. Derzeit für mittel- bis hochqualitative Daten das Mass mit bester Korrelation zum MOS.
- Luminance similarity score (LSS): in 8×8 Pixel Blöcken werden Wechel in der Helligkeit gewichtet summiert.
- Edge similarity score (ESS): in 8×8 Pixel Blöcken werden dominierende Kantenrichtungen verglichen.
- Local entropy: im Wesentlichen wird auf lokaler Basis die Differenz zur Entropie eines zufälligen Bildes bestimmt.

Beurteilung von Verschlüsselungsverfahren III

Original



10.8dB, LSS-1.4,ESS 0.17, SSIM 0.02

10.6dB, LSS-1.4,ESS 0.17, SSIM 0.49

4.1dB, LSS-4.6,ESS 1.0, SSIM 0.02

Beurteilung von Verschlüsselungsverfahren IV

Die gegenwärtig verfügbaren Methoden weisen für tw. verschlüsselte Daten ungenügende Korrelation zur Wahrnehmung und MOS auf. Auch muss berücksichtigt werden, dass es bis zur sufficient encryption / transparent encryption tatsächlich um die Beurteilung der Qualität geht, für content security spielt nur mehr die Frage der Erkennbarkeit von Bildinformation eine Rolle. Das muss anders gemessen werden !

Als mögliches Beispiel kann hier die Biometrie dienen: zur Prüfung von privacy-preserving Verschlüsselungsverfahren für Überwachungsvideos wird nach dem Ausführen von Angriffen ein Gesichtserkennungsverfahren angewendet, um die (automatisierte) Erkennbarkeit zu prüfen. Ebensolches wird im bereich Fingerprint Erkennung auf tw. verschlüsselte Fingerabdrücke angewendet.

Mögliche Strategie auf “generelle Bilder”: im Bereich CBIR gibt es verschiedenste Methoden zur Ähnlichkeitsbeurteilung von Bildern (z.B. SIFT, SURF, etc. Deskriptoren). Solche hochgradig robusten Verfahren könnten für den Zielzweck verwendet werden. Aktuelles Forschungsthema.

Information Hiding & Watermarking

Ein Vorteil der digitalen Medien ist die Möglichkeit, andere Daten in einem Hauptdatenstrom (und nicht in Metadaten eines Formats !) unsichtbar bzw. unhörbar einzubetten ("Information Hiding" [18, 17]). Für diese Technologie gibt es verschiedene Anwendungsbereiche wie

- DRM / Robust 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 (z.B. besondere Disketten-Spuren, DVD). Durch die Welle des elektronischen Kommerzes wird aber nach Software-Lösungen gesucht. Ein mögliches Vorgehen wird als "Digital Watermarking" [5, 9] bezeichnet und soll das Vorhandensein eines nicht wahrnehmbaren digitalen "Wasserzeichens" (mit Copyright Info) beschreiben.

- Steganographie ("verborgenes Schreiben"),

- Manipulationserkennung: Ein weiteres Problem ist die Möglichkeit digitale Daten zu verändern ohne wahrnehmbare Spuren zu hinterlassen. Daher wird ein Intergitätsschutz für digitale Daten benötigt.

Information Hiding & Watermarking: Begriffe

Wenn es um den Schutz und die Verwertung von Copyright geht spricht man von Digital Rights Management (DRM). Man unterscheidet zwischen Watermarking und Fingerprinting. Beim Watermarking werden alle Kopien der Daten gleich behandelt (Copyright- oder Kopierschutz-Informationen).
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 erfolgreicher Angriff besteht also in dem Nachweis dass 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.
Weitere DRM / Robust WM Applikationen sind Annotation Watermarks (Metainformationen, URLs), Copy Control Watermark (DVD CGMS) und Broadcast Monitoring (z.B. Qualitätskontrolle oder Sendeprotokolle).

Watermarking: Anforderungen/Eigenschaften I

Es gibt eine Reihe von unterschiedlichen Anforderungen an Watermarking Systeme, die sich durchaus gegenseitig widersprechen. Für bestimmte Applikationen wird je nach Anforderungen mehr Gewicht auf einige spezielle Eigenschaften gelegt.

- Robustheit: ist die Widerstandsfähigkeit der eingebetteten Daten gegen unbeabsichtigte Trägerdatenveränderung (Formatkonvertierung, Kompression, A/D Wandlung - drucken & scannen) oder nicht-verfahrensspezifische Versuche die eingebetteten Daten zu entfernen oder zu zerstören (Verrauschung, Filterung, Ausschneiden, Angriffstools wie Stirmark und Checkmark). F”ur 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, z.B. bei Vorschaubildern in niedriger Qualität.

- Rekonstruierbarkeit: können die Daten nach dem Auslesen der WM wieder in den Originalzustand gebracht werden (wenn das Original nicht mehr verfügbar ist) ? Wichtig z.B. für medical imaging (lossless !) !

Watermarking: Anforderungen/Eigenschaften II

- Nachweisbarkeit: kann mit statistischen Methoden nachgewiesen werden, dass in den Trägerdaten andere Daten eingebettet sind (Achtung: Wahrnehmbarkeit impliziert nicht notwendigerweise Nachweisbarkeit, z.B. Farbmanipulation).

- Kapazität: Datenmenge die mit einem bestimmten Verfahren eingebettet werden kann. Möglichst hohe Kapazität bei gleichzeitig hoher Robustheit und geringer Wahrnehmbarkeit ist natürlich wünschenswert.

- Sicherheit: selbst bei bekanntem Verfahren soll es für Angreifer nicht möglich sein das Watermark zu entfernen oder zu zerstören. Dies wird durch Verwendung von geheimen Schlüsselmaterial erreicht. Weiters ist damit gewährleistet daß nur der Eigner des Schlüssels das Watermark auslesen kann. Widerstandsfähigkeit gegen bestimmte Attacken wie z.B. den Koalitionsangriff (mehrere unterschiedlich gewatermarkte Bilder werden zur Entfernung des WM genutzt) oder die Invertierbarkeit (bei Doppel- oder Mehrfachmarkierung kann die Reihenfolge nicht festgestellt werden).

Watermarking: Kapazität

Grundsätzlich ist es natürlich wünschenswert, wenn WM Systeme eine möglichst hohe Kapaziät besitzen. Da dies jedoch meist andere (ungünstige) Eigenschaften impliziert (z.B. Wahrnehmbarkeit, Nachweisbarkeit, geringe Robustheit), gibt es auch Methoden, die die Kapazität minimieren um dafür andere Eigenschaften zu optimieren.

- Zero-bit WM: wird meist eingesetzt um maximale Robustheit zu erreichen. Mögliche Szenarien in denen diese Strategie ausreichend ist sind
  - ⋆ Ein mögliches Einsatzszenario von copy protection Systemen ist die Unterscheidung zwischen kommerziellem Content (verschlüsselt und WM), offenem Content (Plaintext und keine WM) und illegitimen Content (Plaintext und WM, also gecrackter kommerzieller Content). Hier genügt die Unterscheidung WM vs. nicht WM, da auch verschlüsselte von unverschlüsselten Daten unterschieden werden können.
  - ⋆ Im Bereich von copyright protection stellt es sich heraus, dass es juristisch irrelevant ist, wenn der Name des copyright holders in die Daten eingebettet ist. Vielmehr muss belegt werden können, dass das Werk vom Author bei einer Autorengesellschaft registriert worden ist. Dies ist wiederum nur eine ja/nein Frage.

- Multi-bit WM: der klassische Ansatz in dem user-IDs, copyright holder Daten, Metadaten u.s.w. eingebettet werden können.

Watermarking: Schüsselmanagement

Es gibt hier (in teilweiser Analogie zur Kryptographie) unterschiedliche Szenarien:

1. Symmetric Watermarking: the same key is used for embedding and extraction of the watermark.

2. Assymetric Watermarking: One key k_1 is used for embedding the data, a second key k_2 is used for extraction of the WM. Without access to k_2, it is impossible to know and remove the watermark.

3. Public Watermarking: anyone can detect and extract the watermark, even without knowing the key (there is no key) and the content of the mark (blind scheme !). In Private Watermarking, the watermark key and content needs to be known for extraction.

4. Zero-Knowledge Watermarking: during detection and extraction, the WM key is not transmitted to the detector, so it cannot be compromised.
   - Option 1: communication between content owner and detector owner prove the presence of the WM.
   - Option 2: detection in the encrypted domain uisng an encrypted key on encrypted content.

Watermarking: Anforderungen/Eigenschaften III

- Vorhandensein des Originalsignals / Watermarks: wird zur Extraktion des Watermarks das Originalsignal nicht benötigt, spricht man von “semi-blind” (oblivious) Verfahren, ansonsten von non-blind Verfahren. 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. Liegt das WM bei der Erkennung auch nicht vor, spricht man von “blind” Verfahren.

- Komplexität: Rechnerischer Aufwand bei der Einbettung/Extraktion. Je nach Applikation kann hier Symmetrie wichtig oder unwichtig sein. Hier spilet auch die Realisierungsmöglichkeit in Hardware eine Rolle.

- Markierungsdomain: wird für die Markierung das Datenmaterial im Rohformat benötigt oder kann das Verfahren auf Bitstreams (z.B. MPEG oder J2K) ohne vorherige Dekodierung angewendet werden.

Watermarking: Application scenarios for (semi-)Blind Schemes

- **Copy-contol**: Soll aus einem Medium die durch ein Watermark eingebettete copy-control Information ausgelesen werden, macht es keinen Sinn das unmarkierte Original im Gerät zu benötigen um das WM auszulesen, da beim Kunden dann bereits eine Version ohne copy-control Information vorliegt.

- **Annotation WM**: Wird an einen Kunden ein verschlüsseltes Medium mit AnnotationsWM verkauft, müsste im Fall der non-oblivious Detection das gleiche medium ohne WM vorliegen um es extrahieren zu können – overhead !

- **Biometrie**: Werden Sampledaten durch den Sensor mit einem WM versehen (um die Authentizität der Daten zu bestätigen und die Wiederverwendbarkeit zu verunmöglichen) und übertragen, ist die Übertragung der nicht-markierten Daten ein Sicherheitsrisiko.

DRM / Robust Watermarking: Methoden

- Amplitude Modulation: hier wird in der spatial domain WM Information durch Veränderung der Pixel Werte eingebettet. Robustheit ist hier relativ gering.
- Patchwork Verfahren: die Daten werden in verschiedene Blöcke aufgeteilt, deren Eigenschaften gesteuert durch die einzubettende WM Information zielgerichtet verändert werden. Geringe Kapazität.
- Spread Spectrum Embedding: in einem Transformationsbereich wird die WM Information in wichtige Koeffizienten eingebettet (ein Signal mit schmalem Spektrum, das WM, wird auf viele Frequenzen verteilt). Wichtige Koeffizienten werden gewählt, weil diese bei Bildmanipulationen erhalten bleiben. Hohe Robustheit, aber geringe Kapazität.
- Quantisierungs-basiertes Embedding: WM Information wird durch Anwendung von verschiedenen Quantisierungsstrategien eingebettet (die WM Information steuert die Wahl des Quantisers), die beim Auslesen identifiziert werden können. Hohe Kapazität, jedoch geringere Robustheit.

WM Information sind hier typischerweise Gauss-verteilte Zufallsfolgen (die WM Information ist z.B. der Seed des Generators) oder bi-polare WM Folgen (aus 0/1 oder -1/1).

DRM / Robust Watermarking: Amplitude Modulation

Die naheliegenste Idee ist es, Daten im Bereich der LSB der unwichtigsten Farbkanäle zu verstecken, da diese die Wahrnehmung kaum beeinträchtigen werden. Allerdings sind solche Verfahren extrem unrobust, da schon durch leichte Kompression die LSB Information völlig verändert wird.

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.

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

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

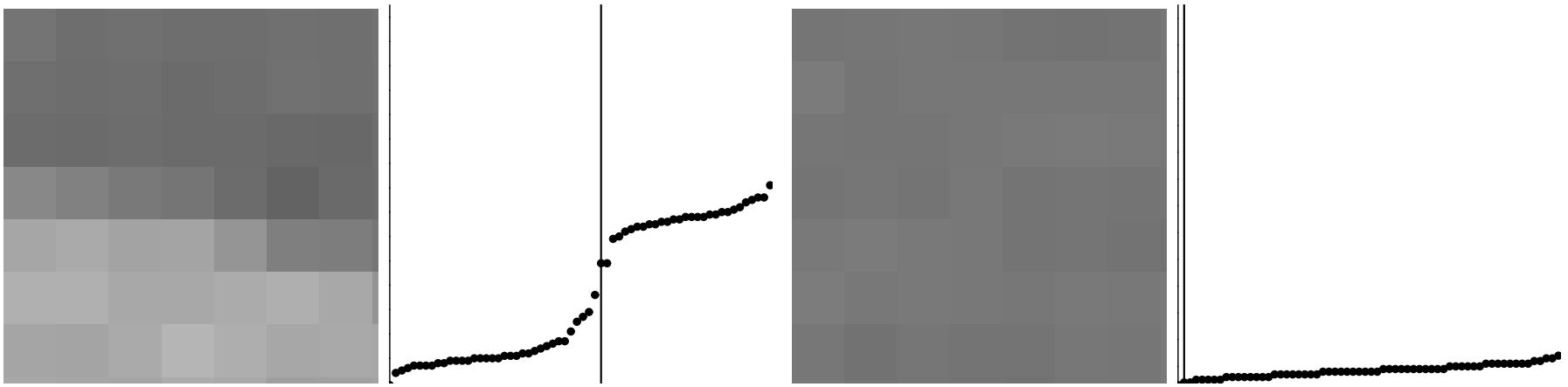
α steuert die Einbettungsstärke. Zum Auslesen der Watermarkbits wird eine sternförmige Umgebung eines Pixel betrachtet und daraus eine Prediction für das zentrale Pixel berechnet. Für jedes Watermarkbit sind nun durch die redundante Einbettung mehrere Predictions und ver”anderte Werte vorhanden was f”ur die blinde (semi-blind & public) Rekonstruktion der WM Information verwendet wird.

DRM / Robust Watermarking: Patchwork Verfahren I

Als Beispiel betrachten wir den Algorithmus von Bruyndoncks: ein Watermark Bitstring wird Bildblöcken eingebettet. Dazu sind folgende Schritte nötig:

- Klassifikation der Pixel eines Blocks in zwei Zonen (mehr oder weniger) homogener Helligkeit.
- Unterteilung jeder Zone in Kategorien die durch ein vorgegebenes Muster definiert werden (kann als Schlüssel interpretiert werden).
- Manipulation der Helligkeitsmittelwerte für jede Kategorie in jeder Zone.

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.



DRM / Robust Watermarking: Patchwork Verfahren II

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. Der Block mit hartem Kontrast kann sofort behandelt werden, während im Block mit Noise Kontrast die Aufteilung so geählt wird, dass die Zonen 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 können von Block zu Block wechseln (Sicherheitskonzept – Schlüssel).

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

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

DRM / Robust Watermarking: Patchwork Verfahren III

Nun sind die Pixel in den Blöcken in 5 Mengen unterteilt: die Weggelassenen (bei zu 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:

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

Nun werden die Watermarkbits wie folgt eingebettet:

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

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

DRM / Robust Watermarking: Patchwork Verfahren IV

Gleichzeitig soll aber der Helligkeitsmittelwert in jeder Zone konstant gehalten werden (nicht-Wahrnehmbarkeit):

$$\frac{n_{1A}l'_{1A}+n_{1B}l'_{1B}}{n_{1A}+n_{1B}}=l_1\,,\text{and}\ \frac{n_{2A}l'_{2A}+n_{2B}l'_{2B}}{n_{2A}+n_{2B}}=l_2$$

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:

$$s''_k=\begin{cases}0 & \text{if } l''_{1A}-l''_{1B}<0 \text{ und } l''_{2A}-l''_{2B}<0\\ 1 & \text{if } l''_{1A}-l''_{1B}>0 \text{ und } l''_{2A}-l''_{2B}>0\end{cases} \tag{4}$$

Nun können Standardmethoden verwendet werden um das rekonstruierte Watermark S'' mit dem Original S zu vergleichen. Die Robustheit dieses Verfahrens ist ebensfalls eher gering, jedoch ist Schlüsselabhängigkeit realisiert. Bei Kenntnis des Schlüssels muss das WM nicht bekannt sein um ausgelesen werden zu können (public WM).

DRM / Robust Watermarking: Spread Spectrum Methoden (im Transformationsbereich) I

Das Wasserzeichen besteht typischerweise aus einer normalverteilten Folge von n Zahlen s_i mit Mittelwert 0 und Varianz 1. Um nun ein Element der Watermarkfolge s_i in einen Koeffizienten c_i einzubetten wird wie folgt vorgegangen (wobei c_i' der gewatermarkte Koeffizient ist und α die Einbettungsstärke ist):

$$c_i' = c_i(1 + \alpha s_i) \tag{5}$$

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):

DRM / Robust Watermarking: Spread Spectrum Methoden (im Transformationsbereich) II

$$s_i^* = \frac{c_i^* - c_i}{\alpha c_i} \tag{6}$$

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.

Das Verfahren von Cox I

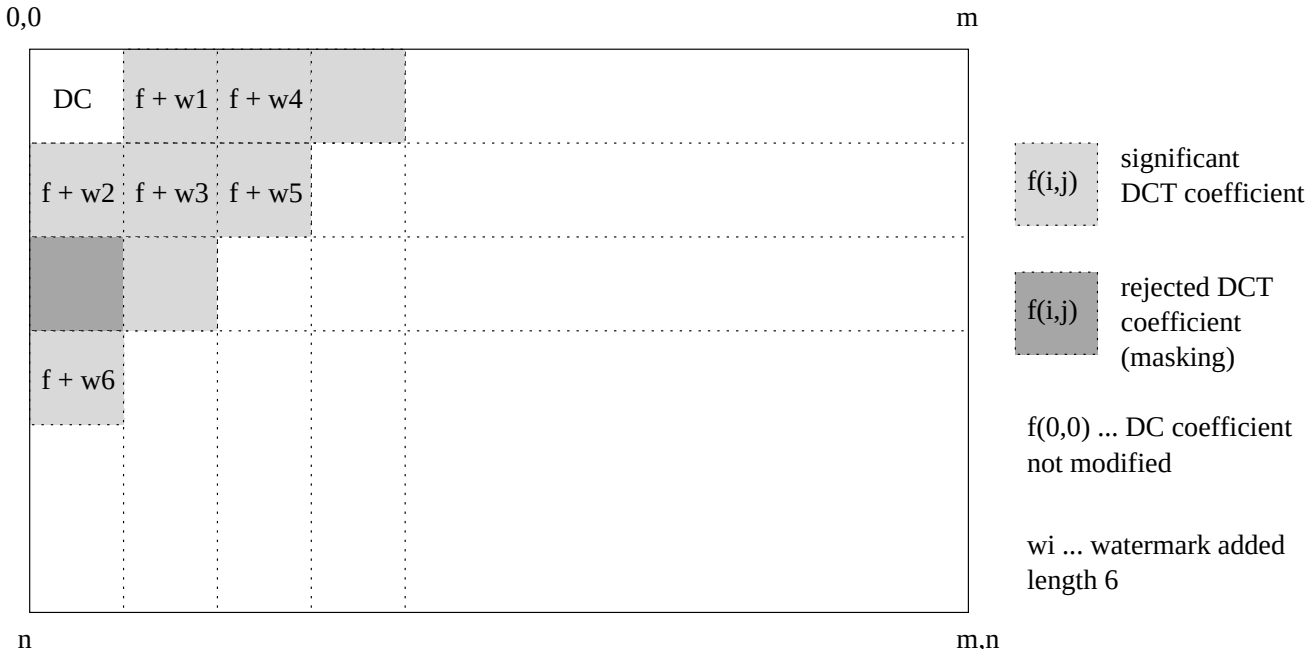
DER klassische Algorithmus zur Einbettung von copyright-schützenden Watermarks ist der von Cox (NEC Research, MIT): zu den 1000 grössten DCT-Koeffizienten $f(m,n)$ wird eine eben so lange um 0 normalverteilte Folge von Zufallszahlen w_i addiert:

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

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.

Daher werden die grossen Koeffizienten verwendet, da diese die meiste Energie auf sich konzentrieren und die wesentlichen Träger der zentralen Bildinformation sind. Solange die Bildinformation in den wesentlichen Aspekten unberührt bleibt, sollten auch diese Koeffizienten sehr ähnlich bleiben.

Das Verfahren von Cox II



Dieses Verfahren zeichnet sich durch sehr hohe Robustheit aus. Soll es als oblivious Verfahren realisiert werden, wird das WM Erkennungsproblem als Signal Detection in Noise formuliert (die Trägerdaten = das Bild werden dabei als Noise interpretiert). Problematisch ist hier mögliche Host / Signal Interference. Daher ist es bei diesen Methoden von zentraler Bedeutung, die statistische Verteilung der Host Daten so gut wie möglich zu modellieren, um zuverlässige Detektoren designen zu können (z.B. Generalized Gaussian, Laplace Verteilung für Koeffizienten). Nur für normalverteilte Daten ist Korrelation die optimale Wahl (eventuell: DC Koeffizienten oder LL Wavelet Subband.

Wavelet Spread Spectrum Verfahren I

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.

Wavelet Spread Spectrum Verfahren II

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:

Wavelet Spread Spectrum Verfahren III

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.

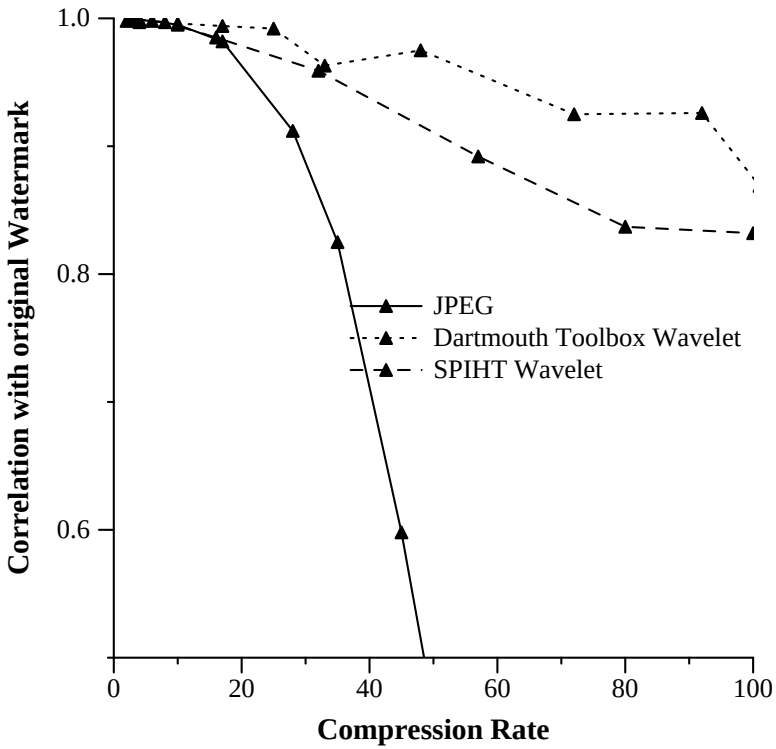
3. $T_{k,l}$ wird halbiert.

4. Sind noch Watermarkelemente nicht eingebettet, nochmals iterieren.

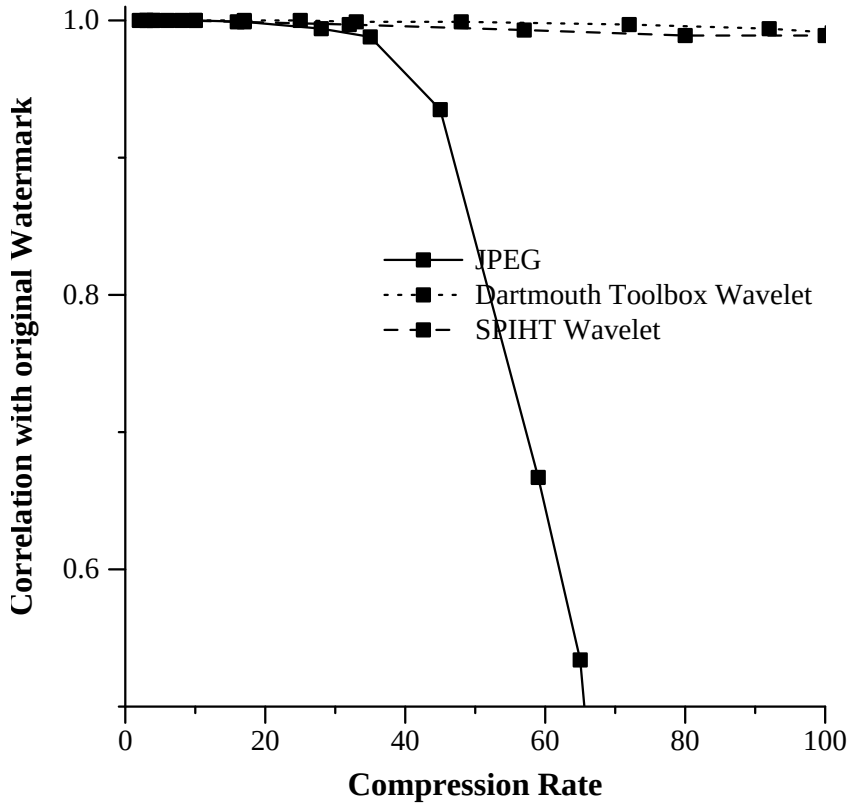
Für das tatsächliche Einbetten wird die folgende Formel verwendet:

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

Spread Spectrum Verfahren : Robustheit



Watermark cox(0.0475) under compression



Watermark wang(0.275) under compression

Robustheit gegenüber Kompression hängt also wesentlich vom verwendeten Verfahren ab, kann aber sehr hoch sein. Kontroversiell diskutiert wird in der Literatur ob es gut ist, die gleiche Strategie (z.B. Transformation, Begriff für Signifikanz) für Kompression und Watermarking zu verwenden oder nicht.

DRM / Robust WM: Quantisierungs WM I

Diese Verfahren gehören zu einer grösseren Klasse von Verfahren, bei denen der Wertebereich des Signals in verschiedene Teilbereiche aufgeteilt wird, die von einer Funktion $Q()$ auf die Menge der einzubettenden Werte abgebildet wird (z.B. auf $b \in \{0, 1\}$ für binäre WM Information).

Seien I_0 die Originaldaten und I_1 das markierte Signal, so wird I_1 aus dem Teilbereich des Signals der auf b abgebildet wird so ausgewählt, dass $b = Q(I_1)$ gilt (das ist dann der Ausleseprozess). Zusätzlich wird I_1 "so nahe wie möglich" zu I_0 gewählt um einen möglichst geringen Impact auf die Wahrnehmung zu erzielen.

Dies ist anders als beim spread spectrum Ansatz: dort wird die WM Information b additiv oder multiplikativ in die Daten eingebracht; um ein Bit einzubetten, ist der Unterschied zwischen I_0 und I_1 eine Funktion von b, d.h. $I_1 - I_0 = f(b)$. Hier spielt also im Gegensatz zum obigen Ansatz I_0 eine Rolle und die Kentnnis von I_0 verbessert die WM Extraktion.

Quantisierungs-basiertes WM ist demnach intrinsisch oblivious.

DRM / Robust WM: Quantisierungs WM II

Es gibt viele Möglichkeiten um den Signalbereich in geeignete Bereiche aufzuteilen. Eine Möglichkeit ist es, bestimmte n-Tupel von Koeffizienten zu betrachten. Die Abbildung $Q()$ kann auf die Grössenrelation innerhalb der n-Tupel angewendet werden.

Ein klassisches Verfahren ist das von Koch []: es werden in Falle einer block-basierten DCT aus jedem Block zwei Koeffizienten v_1, v_2 ausgewählt (nach perzeptuellen Kriterien) aus denen die markierten Koeffizienten v_1', v_2' so gewählt werden, dass $v_1' > v_2'$ für $b = 1$ und $v_1' < v_2'$ für $b = 0$. Wesentlich ist es, dass v_1, v_2 und v_1', v_2' perzeptuell ähnlich sind. Ebenso können auf Tripel von Koeffizienten solche Relationen abgebildet werden oder nur auf das Vorzeichen von Daten.

Um ein Mindestmass an Robusteheit zu erreichen, muss der Einbettungsvorgang eine Toleranzzone mit der Grösse A erzwingen, sodass $|v_1' - v_2'| \geq A$ (für Ordnungsrelation zwischen zwei Datenwerten) bzw. $|v'| \geq A$ (Vorzeichen eines Wertes).

Ein sehr einfaches Bsp. ist auch sog. “Even-Odd Embedding”, wo die dem Sample am nächsten liegende gerade Zahl verwendet wird, um eine “0” einzubetten, und die nächstgelegene ungerade Zahl um eine “1” einzubetten. Auslesen ist einfacher Even/Odd Check, Toleranzgrenze ist 1 symmetrisch um jede ganze Zahl. Sehr ähnlich dem LSB Einbetten !

Quantisation Index Modulation (QIM) I

QIM verallgemeinert die Idee des Even-Odd Einbetten, indem mehrere Quantiser definiert werden. Beim Einbetten entscheidet der einzubettende WM Wert, welcher der Quantiser für einen Datenwert angwendet wird. Beim Auslesen wird der nächstgelegene Quantiser Rekonstruktionspunkt gesucht: der zugeörige Quantiser bestimmt die ausgelesene WM Information.

Wir betrachten im Folgenden als einfaches Bsp. einen skalaren Quantiser (im konkreten Bsp. dann mit Quantisierungs- Schrittweite $q = 2$):

$$Q(x) = q\lfloor\frac{x}{q}\rfloor \text{ oder im symmetrischen Fall} Q(x) = q\lfloor\frac{x+q/2}{q}\rfloor$$

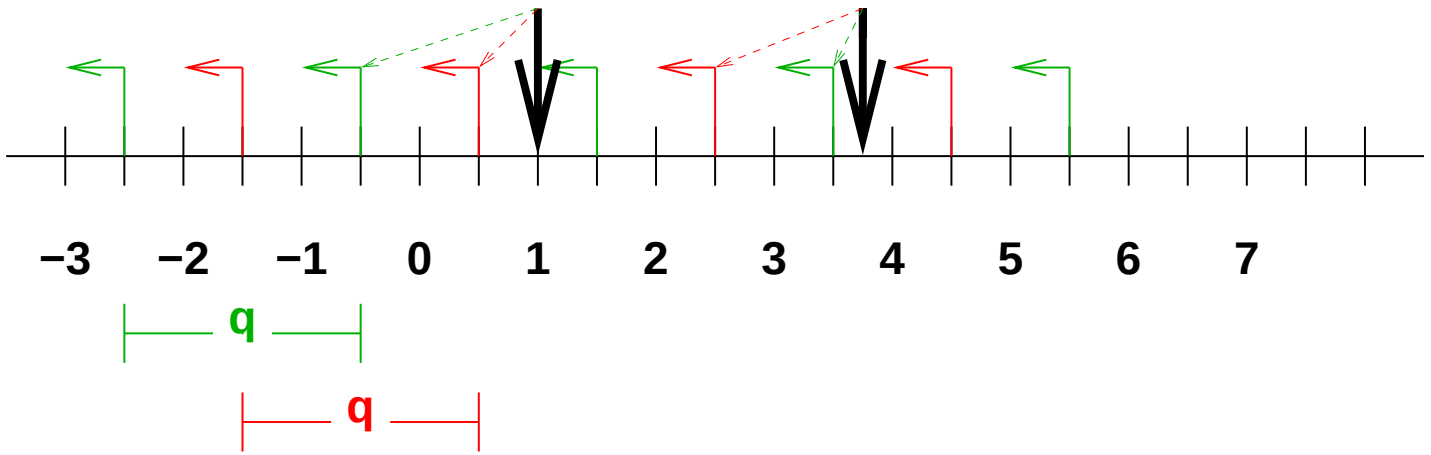
Wir betrachten im Folgenden den einfachere nicht-symmetrischen Fall und definieren für QIM zwei Quantiser wie folgt ($+-q/4$ wird als dither vector bezeichnet):

$$Q_0(x) = q\lfloor\frac{x+q/4}{q}\rfloor - q/4 \text{ bzw.} Q_1(x) = q\lfloor\frac{x-q/4}{q}\rfloor + q/4$$

QIM II

Setzen wir nun $q = 2$ erhält man:

$$Q_0(x) = 2\lfloor\frac{x+1/2}{2}\rfloor - 1/2 \text{ bzw. } Q_1(x) = 2\lfloor\frac{x-1/2}{2}\rfloor + 1/2$$

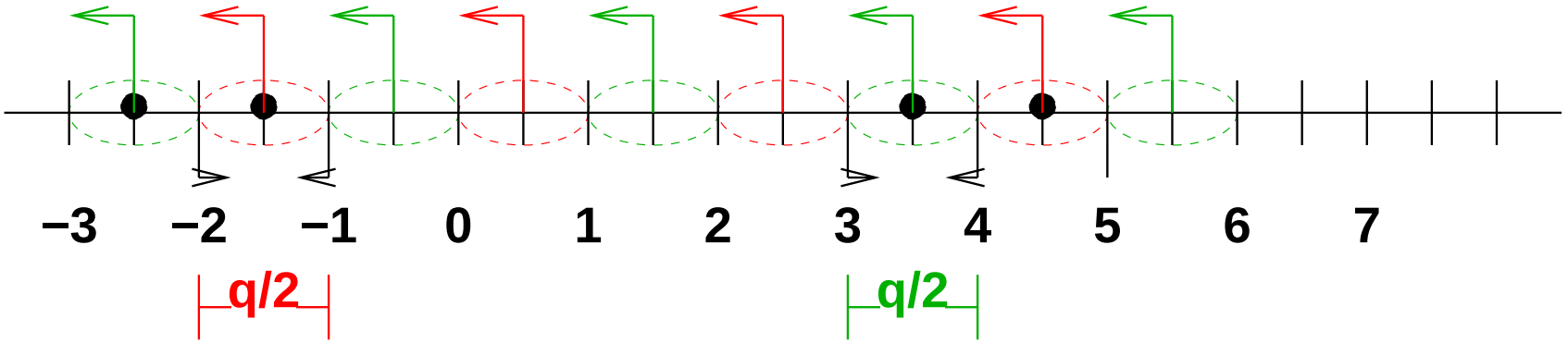


Betrachten wir also zuerst Q_0 (den grünen Quantiser): $Q_0(0) = -1/2$, $Q_0(1) = -1/2$, $Q_0(2) = 11/2$, $Q_0(3) = 11/2$, $Q_0(4) = 31/2$, $Q_0(5) = 31/2$, u.s.w., d.h. in den grünen Intervallen werden alle Werte auf den linken Eckpunt des grünen Intervalls quantisiert. Für Q_1 (den roten Quantiser) gilt analog: $Q_1(0) = -11/2$, $Q_1(1) = 1/2$, $Q_1(2) = 1/2$, $Q_1(3) = 21/2$, $Q_1(4) = 21/2$, $Q_1(5) = 41/2$, u.s.w., d.h. auch in den roten Intervallen werden alle Werte auf den linken Eckpunt des roten Intervalls quantisiert.

Je nach einzubettendem WM Bit wird nun Q_0 oder Q_1 auf einen Datenpunkt (grosser schwarzer Pfeil) angewendet.

QIM III

Auslesen der WM Information geschieht wie folgt: es wird derjenige Quantiser Rekonstruktionspunkt gewählt, der dem ausgelesenen Datenpunkt am nächsten liegt. Der für diesen Rekonstruktionspunkt "zuständige" Quantiser Q_i definiert das ausgelesene Bit.



Falls es zu keiner Modifikation von I_1 gekommen ist, kann das WM ohne Störung ausgelesen werden. Im Fall einer Störung liegt um jeden Rekonstruktionspunkt eine Toleranzzone der Grösse $d/2$. Hier sieht man den Tradeoff zwischen Robustheit und Wahrnehmenbarkeit besonders schön: je grösser die Quantiser Stepsize, desto grösser die Toleranzzone, aber desto stärker die Quantisierung.

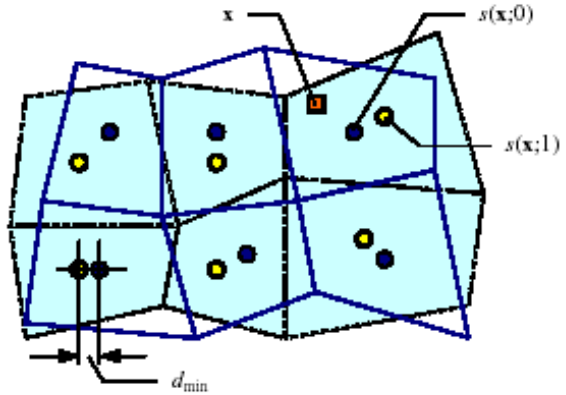
QIM IV

Diese Idee kann generell angewendet werden auf beliebige Quantiser $Q_m(s)$, $m \in \{0,1\}$, where each Q_m eine Abbildung der reellen Achse in ein Codebuch $B_m = \{b_{1,m}, b_{2,m}, b_{3,m}, b_{4,m}, \cdots, b_{L,m}\}$ ist. Die Ausgabewerte der Quantiser $b_{i,m}$ $1 \leq i \leq L$ werden als Rekonstruktionspunkte bezeichnet.

Wird im Empfänger das Sample y des Signals I_1 erhalten, wird das WM Bit bestimmt durch:

$$m(y) = argmin_{m \in \{0,1\}} |y - b_m|$$

Diese Idee kann natürlich auch für Vektorquantisierung, in verschiedenen Transformationsbereichen und in weiteren Variationen angewendet werden.



Watermark Security: Applikationsszenarien

1. Eine Werbefirma bettet ein Watermark in ihre Werbespots ein und zeichnet die Sendungen der Broadcaster auf um die WMs identifizieren zu können. Die Rechnungen der Broadcaster können mit den Logs verglichen werden. Ein Broadcaster will nun selbst einen eigenen Werbespot senden, möchte aber dass die Werbefirma dafür bezahlt. Er bettet also das identische Watermark in seinen Spot ein und sendet diesen anstelle des Spots der Firma. *Unauthorized embedding* oder *forgery attack* oder *copy attack*.
2. Ein Watermarking Service A bietet das Einfügen von WMs und eine entsprechende Suche nach gemarkten Bildern im Internet an. Dies soll den Kunden das Auffinden von nicht autorisierten Kopien ermöglichen. Das Konkurrenzunternehmen B bietet nun einen billigeren (ohne Einbetten) Service an, der die von A eingebetteten WMs sucht und die Ergebnisse sammelt. *Unauthorized detection* oder *passive attack*.
3. Ein Filmstudio bettet vor der Distribution der Filme copy-control WMs ein mit dem Ziel dass alle digitalen Rekorder WM Detektoren beinhalten und diese Filme dann nicht kopieren werden. Ein Video Pirat kann ein Device entwickeln mit dem er diese WMs entfernt und kann dann illegale Kopien machen (wie der Plan bei DVD). *Unauthorized removal*. (Anmerkung: wenn ein Copy-control System ein WM braucht dass kopiert werden darf, müsste es sicher gegen unauthorized embedding sein !)

Watermark Security: Unauthorized Embedding I

- Copy Attack: identisches WM wird auf andere Daten übertragen
- Im allgemeineren Fall wird eine neue Nachricht als WM Kodiert und eingebettet

Wird das WM vor dem Einbetten mit einem geheimen Schlüssel verschlüsselt, so kann die zweite Art des Angriffs verhindert werden, nicht jedoch die Copy Attack, weil ja das verschlüsselte WM genauso kopiert werden kann. Um diese Form des Angriffs zu verhindern, muss eine Beziehung zwischen WM und Cover Daten geschaffen werden, beispielsweise könnte das WM eine Signatur des Trägerbildes enthalten. Problem: durch das Einbetten wird das Bild verändert, sodass die Signatur nicht mehr gültig wäre. Hier werden robuste Bildfeatures benötigt, die invariant bzgl. des WM Einbettens sind (z.B. niederfrequente DCT Koeffizienten).

Watermark Security: Unauthorized Embedding II

Vorgehensweise:

1. Eine Beschreibung der Trägerdaten durch robuste Features wird berechnet (vgl. robuste Multimedia Hashes).

2. Das WM wird mit der Beschreibung konkatiniert und ein klassicher Hash wird davon berechnet.

3. Dieser Hash wird mit einem private Key verschlüsselt und ergibt damit eine Signatur.

4. Das WM und die Signatur wird eingebettet. Das Einbetten darf die robusten Features nicht verändern.

Watermark Security: Unauthorized Detection & Unauthorized Removal

Unauthorized Detection kann sich auf das reine Erkennen von eingebetteten Daten beschränken (was ein rein steganographisches Problem ist) oder auch die Dekodierung der WM Information beinhalten (was durch Anwendung von klassischer Verschlüsselung der WM Information verhindert werden kann). Zwischen diesen beiden Extremfällen liegt der Fall dass der Angreifer zwischen verschiedenen WMs unterscheiden kann.

Unauthorized Removal ist erfolgreich, wenn das WM nicht mehr ausgelesen werden kann, das Trägerbild aber für den Angreifer ausreichende Qualität hat. Es wird unterschieden zwischen *elimination attacks* und *masking attacks* (wie z.B. den Synchronisationsattacken durch leichte Rotation des Trägerbildes). In [8] wird durch die Verwendung von geheimen Transformationsdomains (durch Wavelet Filterparametrisierung) beim Einbetten sowohl unauthorized detection als auch unauthorized removal erfolgreich verhindert.

Watermark Security: weitere Angriffe I

- Scrambling Attacke: ist ein Angriff auf höherer Ebene bei dem Teile der Trägerdaten verschlüsselt vorliegen sodass ein WM Detektor wirkungslos ist. Es muss dafür gesorgt werden, dass das Scrambling vor der Verwendung invertiert wird. Eine Permutation von 8x8 Blöcken wird allerdings bei WM Verfahren die unabhängig gerade solche Blöcke bearbeiten nicht ausreichen. Kann zur Umgehung eines Copycontrol Mechanismus benutzt werden. Der Input in das Aufzeichnungsgerät wird verschlüsselt sodass die WM nicht entdeckt wird. Beim Dekodierungsvorgang wird das Signal vor dem Viewer entschlüsselt. Kann zum Umsetzen von PG sogar argumentiert werden.

- Mosaic Attacke: eine Spezialform bei der die Bilder in kleine Einheiten zerteilt werden die für eine WM Detektion zu klein sind. Beim Display werden diese Teile dann in Tabellenform dargestellt sodass kein Unterschied zum Original sichtbar ist. Wurde v.a. für Web-browser angedacht.

Watermark Security: weitere Angriffe II

Eine *falsch positive* Detektion liegt vor wenn ein WM angezeigt wird obwohl kein WM eingebettet wurde. Wenn das Bild bereits ein anderes WM beinhaltet wird dies als *WM Kollision* bezeichnet. Beides führt beim Feststellen des Eigentümers zu Problemen und stellt die Zuverlässigkeit von WM Systemen in Frage.

Die *Ambiguity attack* [6] beruht auf zwei in einem Bild einbettbaren WMs. Die rechtmäßige Bildeigentümerin Alice generiert aus ihrem Originalbild I das gewatermarkte Bild $\tilde{I} = I + w$ wobei w ihr WM ist. $\tilde{I}$ wird an die Kunden weitergegeben. Wird ein Bild I' auf Copyrightinformation untersucht, so berechnet Alice $I' - I = x$, anschließend wird die Korrelation zwischen x und dem ursprünglich eingebetteten WM w berechnet: $c(w,x)$.

Mallory will nun seinerseits behaupten, das Bild gehöre ihm. Er berechnet $I' = \tilde{I} - x = I + w - x$ und behauptet, I' sei das Original. Alice kann ihr WM in Mallory's Bild nachweisen: $I' - I = w - x$, $c(w, w-x) = 1$ für ein zufälliges Muster x. Da zwei WMs eingebettet werden können, stört die Subtraktion kaum. Andererseits kann aber Mallory sein WM im Original von Alice nachweisen ($I - I' = x - w$, $c(x-w,x) = 1$) und auch in den von Alice verbreiteten Kopien ($\tilde{I} - I' = I + w - (I + w - x) = x$, $c(x,x) = 1$). Die Attacke beruht auf der Invertierbarkeit des WM Schemas. Die kann z.B. erreicht werden indem die WM Einbettung eine one-way function des Originalbildes ist.

Watermark Security: weitere Angriffe III

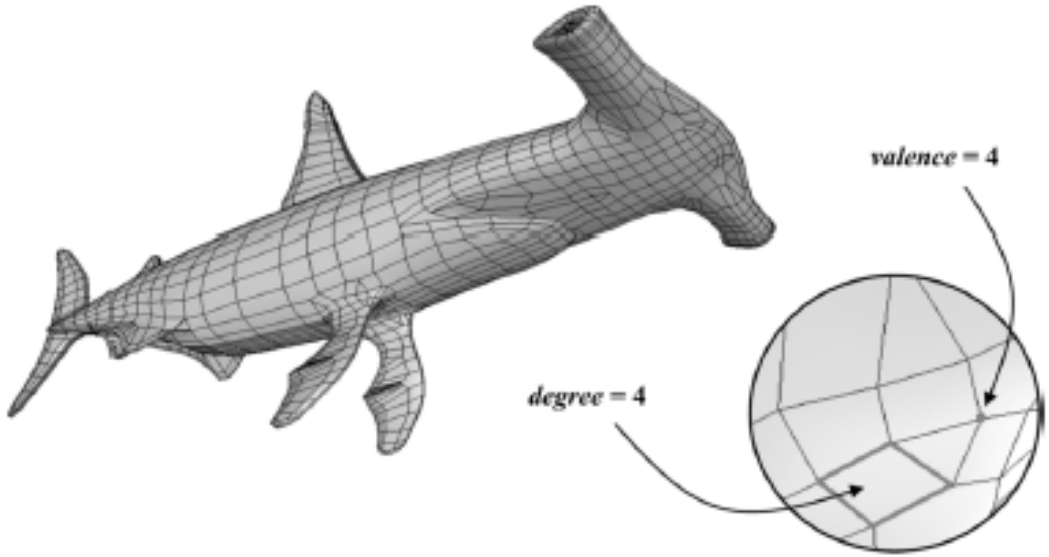
- Oracle/Sensitivity Attacke: geht von der öffentlichen Verfügbarkeit eines WM Detektors mit binärem Output aus. Der Angriff beginnt mit der Konstruktion eines Bildes das nah an der Schranke des Detektors liegt. Dann wird systematisch die Sensitivität des Verfahrens auf Modifikation einzelner Pixel untersucht, sodass schlußendlich bestimmt werden kann, welche Modifikationen den geringsten Einfluß auf die Wahrnehmbarkeit und den größten Einfluss auf die Nachweisbarkeit haben.

- Gradienten Attacke: geht von der öffentlichen Verfügbarkeit eines WM Detektors mit Ausgabe der Detektionswerte aus. Gradientenverfahren werden angewendet um die optimalen Modifikationen (wie vorher) zu bestimmen.

Man sieht also dass die öffentliche Verfügbarkeit von WM Detektoren problematisch ist.

Robustes WM für 3D Graphik Daten: 3D-Meshes I

In Analogie zur partiellen Verschlüsselung für 3D Graphik Daten betrachten wir nun auch “the second line of defence” in DRM, robustes WM für 3D Meshes (da 3D Meshes die allgemeinste Darstellung von 3D Graphikdaten sind in die andere Varianten übergeführt werden können, gibt es auch kaum andere Literatur). Es gibt wesentlich mehr Literatur als zur Verschlüsselung von Meshes, jedoch auch wesentlich weniger als im Bereich von Audio, Bild- und Video WM.



Die Valenz einer Vertex gibt die Anzahl der Kanten (edges) an, die von ihm ausgehen (oder äquivalent die Anzahl der direkten Nachbarn), der Grad (degree) einer Fläche (facet) gibt die Anzahl der begrenzenden Kanten an.

Robustes WM für 3D Graphik Daten: 3D-Meshes II

Die Gründe für die geringere Anzahl an Verfahren verglichen zu “klassichen” WM Medien sind einerseits die Schwierigkeiten durch die freiere Topologie und das irreguläre Sampling von 3D Meshes und andererseits die vielfältigeren Angriffsmöglichkeiten gegen solche Verfahren. WM-relevante Unterschiede zu klassichen Medien sind:

- Bei 3D Meshes gibt es keine offensichtliche intrinische Reihenfolge der Daten (z.B. vertices und facets), diese können in vielen Varianten umgeordnet werden. In klassischen Medien wird die Reihenfolge der WM-Informationsträger zur Synchronisation zwischen einbetten und auslesen benutzt.

- Durch das irreguläre Sampling ist es wesentlich schwieriger, Spektralanalyse über Transformationen wie DCT, DWT, Fourier etc. durchzuführen. In klassischen Medien sind diese Transformationen eines der wesentlichen WM Tools.

- Ein 3D Objekt kann durch verschiedenste Arten von Geometrie- und Connectivity Informationen dargestellt werden, z.B. ein Mesh mit 1-to-4 connectivity (ein vertex hat 4 Nachbarn) oder mit 1-to-6 connectivity. Ein robustes WM sollte in allen möglichen Darstellungen erhalten bleiben.

Angriffe gegen 3D-Mesh WM I

- Angriffe gegen die Geometrie (Vertex Positionen)
  - ⋆ Ähnlichkeits Transformationen: Translation, Rotation, Skalierung und Kombinationen dieser Operationen. Gegenstrategien sind WM-primitives (Daten in die WM eingebettet werden), die invariant geg. solchen Operationen sind (z.B. Verhältnis zwischen Höhe und Kantenlänge bei Dreiecken), insensitive Sychronisationsschemata (spezifizierte Scanordnung bei Dreiecksseiten bezüglich des Längenverhältnisses), WM in einem Raum der invariant geg. solchen Operationen ist (z.B. normierte sphärische Koordinaten) und Registrierung bzgl. des WM-freien Originals (hier werden die Verfahren notwendigerweise non-blind, aber hohe Robustheit wird erzielt).
  - ⋆ Signalverarbeitungs Attacken: Random Noise Addition, Glättung, Kompression / Quantisierung. Ebenso wie der erste Punkt sind das gängige Operationen im Bereich der Animation und der special effects und können WM leicht entfernen. Da diese Angriffe als Modifikation im Bereich der höheren Frequenzen modelliert werden können, ist eine offensichtliche Gegenstrategie die WM Einbettung im Bereich der niederen und mittleren Frequenzen (was auch immer das bei meshes heisst). Auch statistische mesh-features können hier robust sein.
  - ⋆ Lokale Deformation: kann zu schweren Synchronisationsschwierigkeiten führen. Gegenmassnahme ist eine Segmentierung in mehrere Teilmeshes und das redundante Einbetten des WM. Blinde Segmentierung ist kaum zu realisieren.

Angriffe gegen 3D-Mesh WM II

- Angriffe gegen die connectivity information
  - ⋆ Cropping: hat viel mit lokaler Deformations zu tun, Abhilfe schafft Segmentierung in Submeshes und redundantes Einbetten.
  - ⋆ Remeshing & Simplification: hier sind nur solche Methoden robust, die WM primitives verwenden, die die grobe Form des mesh reprs̈entieren. Histogramm-basierte methoden sind hier oft erfolgreich, sind aber auch bei non-uniform remeshing oder subdivision anfällig. Hier werden Restaurierungsmethoden verwendet, die durch remeshing die gleiche connectivity wie das originalmesh erreichen.
- Andere Angriffe: File Attack (Umordnen der vertices oder facets im Beschreibungsfile) und Repräsentationskonvertierung (Approximation durch NURBS Modell oder durch Voxel, das Mesh existiert nicht mehr). Bisher keine in dieser Hinsicht robusten Verfahren.

Transformationen für 3D-Meshes: Spektralanalyse I

Da es nicht möglich ist, eine Transformation direkt auf vertex oder connectivity information anzuwenden, wird die sogenannte mesh Laplace Matrix verwendet, die sowohl geometry als auch connectivity information beinhaltet. Meist wird die "combinatorial Laplacian" oder Kirchhoff Matrix K gebildet.

$$K = D - A$$

wobei D eine Diagonalmatrix ist, deren Diagonalelemente die Valenz von vertex i (also die Anzahl der direkten Nachbarn) sind und A eine Nachbarschaftsmatrix ist, deren Elemente $a_{ij} = 1$ wenn vertex i und vertex j direkte Nachbarn sind und $a_{ij} = 0$ sonst. Ein mesh mit n vertices produziert eine Matrix K mit $n \times n$ Elementen. Im Folgenden wird eine Eigenwert Zerlegung durchgeführt mit

$$K = Q^t \ \Omega \ Q$$

wobei Ω die Diagonalmatrix mit n Eigenwerten als Diagonalelementen ist, und Q die Eigenvektor Matrix bestehend aus n Eigenvektoren w_i. Die Eigenwert / Eigenvektor Paare werden nach aufsteigend nach der Grösse der Eigenwerte sortiert (entsprechend aufsteigender Frequenz).

Nach der Sortierung wird jede Komponente der kartesischen Vertexkoordinaten $v_i = (x_i, y_i, z_i)$, $1 \leq i \leq n$ separat of den i-ten normierten Eigenvektor $e_i = \frac{w_i}{|w_i|}$ projeziert, was n Vektoren von mesh Spektralkoeffizienten $r_i = (r_{i,x}, r_{i,y}, r_{i,z})$ generiert.

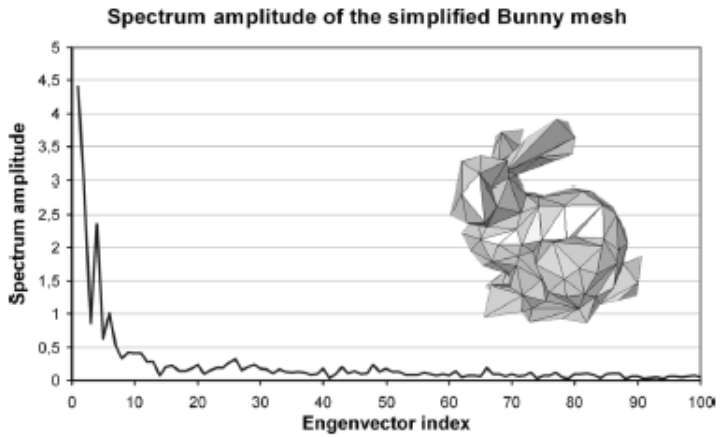
Spektrale Transformationen für 3D-Meshes: Spektralanalyse II

Damit erhält man folgende Darstellung in Analogie zur klassischen Freuqenzanalyse:

$$(x_1, x_2, \ldots, x_n)^T = r_{1,x}e_1 + r_{2,x}e_2 + \ldots + r_{n,x}e_n$$

$$(y_1, y_2, \ldots, y_n)^T = r_{1,y}e_1 + r_{2,y}e_2 + \ldots + r_{n,y}e_n$$

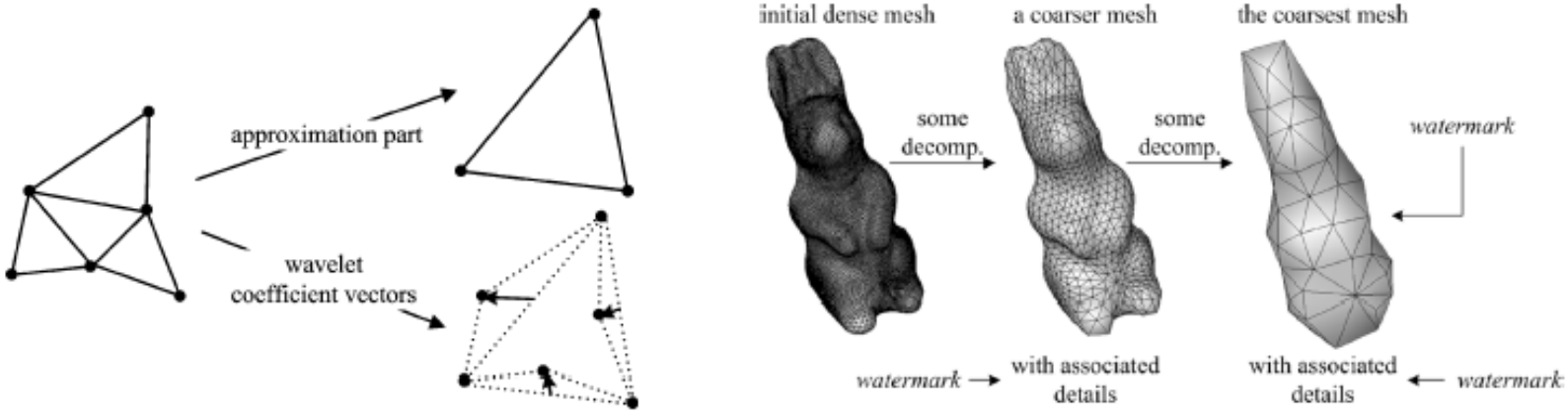
$$(z_1, z_2, \ldots, z_n)^T = r_{1,z}e_1 + r_{2,z}e_2 + \ldots + r_{n,z}e_n$$



Auch die Darstellung des Spektralbereichs erinnert von der Form (grosse Amplitude für niedere Frequenzen) stark an ein Powerspektrum. Die Probleme mit dieser Darstellung sind einerseits der hohe Rechneaufwand für die Eigenwertzerlegung (K ist eine grosse Matrix ! Abhilfe: Submeshes und effiziente Algorithmen) und andererseits die Abhängigkeit von der connectivity information (Abhilfe: remeshing analog zum vorhandenen Original).

Spektrale Transformationen für 3D-Meshes: Wavelets

Wavelet Analyse von 3D meshes wird durch die “lazy wavelet decomposition” erreicht: im Beispiel werden 4 Dreiecke in eines vereinigt, nur drei der ursprünglich sech vertices bleiben in der geringeren Auflösung erhalten. Wavelet Koeffizienten werden berechnet als Prediction Fehler der verlorenen vertices, d.h. diese Koeffizienten sind 3-D Vektoren die mit jeder Kante des mesh mit geringerer Auflösung assoziiert sind. Eine gewisse mesh-Regularität ist Vorraussetzung für dieses Verfahren.



In Analogie zur spektralen Zerlegung können WM Informationen in verschiedene “Frequenzen” (hier Auflösungsstufen) eingebettet werden, mit verschiedenen Implikationen für die Robustheit. Ähnliches wurde auch für progressive meshes entwickelt.

Spektral Transformations-basiertes WM für 3D-Meshes I

Es wird vorgeschlagen, die untere Frequenzhälfte der Spektralkoeffizienten zum Einbetten zu verwenden, da diese eine gute Approximation an die Form des mesh darstellen. Es wird durch Wiederholung eine ausreichend lange Folge von WM-bits $b_i \in \{-1,1\}$ erzeugt, zusätzlich wird noch ein WM-key $p_i \in \{-1,1\}$ verwendet (z.B. durch PRNG generiert). WM Einbettung geschieht in jede Koordinate des Frequenzvektors $r_i = (r_{i,x}, r_{i,y}, r_{i,z})$ durch:

$$\hat{r}_{i,x} = r_{i,x} + b_i p_i \alpha$$

mit α die Einbettungsstärke. Abschliessend wird noch das gewatermarkte Mesh rekonstruiert durch:

$$(\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n)^T = \hat{r}_{1,x} e_1 + \hat{r}_{2,x} e_2 + \ldots + r_{n,x} e_n$$
$$(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n)^T = \hat{r}_{1,y} e_1 + \hat{r}_{2,y} e_2 + \ldots + r_{n,y} e_n$$
$$(\hat{z}_1, \hat{z}_2, \ldots, \hat{z}_n)^T = \hat{r}_{1,z} e_1 + \hat{r}_{2,z} e_2 + \ldots + r_{n,z} e_n$$

Spektral Transformations-basiertes WM für 3D-Meshes II

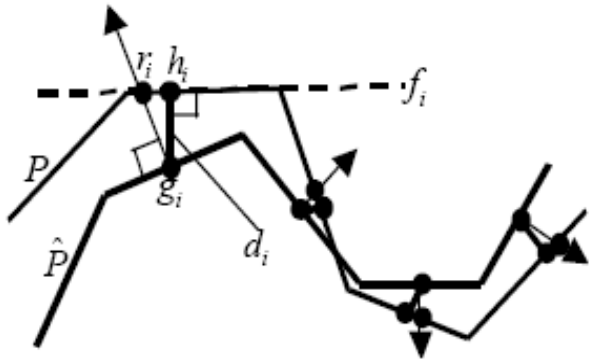
Zum Auslesen des WM wird (wie beim typischen spread spectrum WM) das Originalmesh benötigt. Beide meshes werden einer Spektraltransformation unterzogen und die Referenzkoeffizienten $r_i = (r_{i,x}, r_{i,y}, r_{i,z})$ als auch die (potentiell) gewatermarkten Koeffizienten $\hat{r}_i = (\hat{r}_{i,x}, \hat{r}_{i,y}, \hat{r}_{i,z})$ werden berechnet.

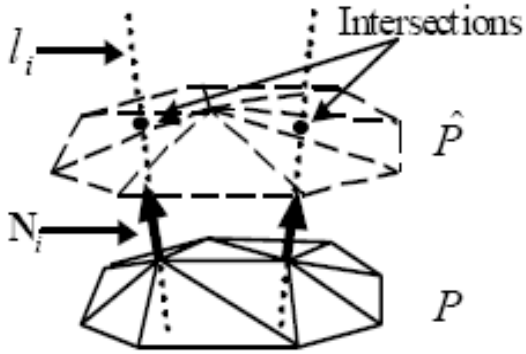
Anschliessend wird die Differenz $(r_i - \hat{r}_i)$ mit dem embedding WM-key $p_i \in \{-1, 1\}$ multipliziert. Das Ergebnis wird dann mit dem origianlen WM korreliert, wobei durch die Einbettung in die drei Koordinatenachsen zusätzliche Redundanz erzielt wird.

Da die Berechnung der Spektralkoeffizienten direkt von den kartesischen Koordinaten der meshes abhängt, ist ein korrektes Alignment bzw. Remeshing der beiden notwendig, um das WM symchronisiert auslesen zu können. Das geht wie folgt:

Transformations-basiertes WM für 3D-Meshes III

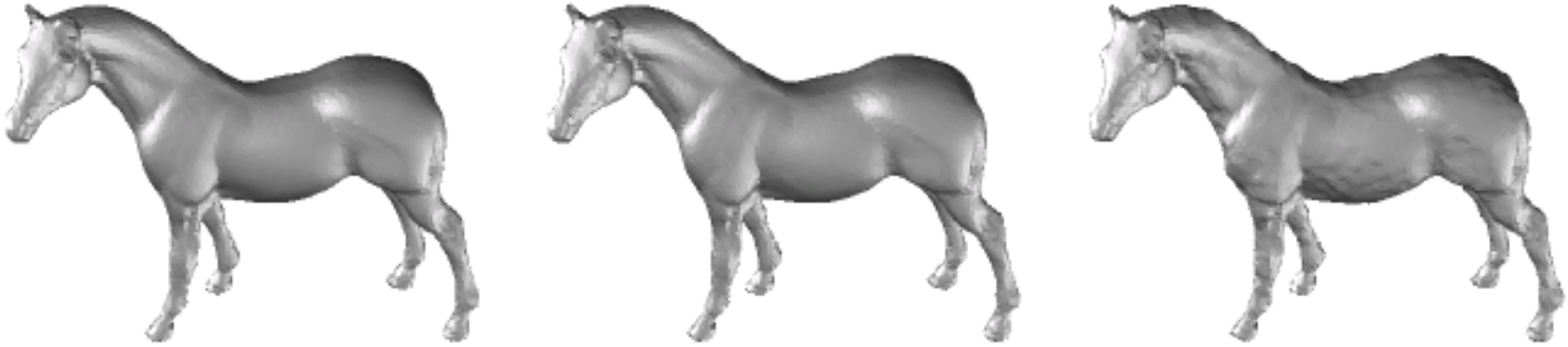
Nach einem groben alignment (manuell oder automatisiert über die Achsen von Punktmengen die auf dem mesh zufällig verteilt werden) wird eine Feinadjustierung vorgenommen über die iterative Minimierung der Summe der Distanzen zwischen original und WM mesh (P und $\hat{P}$). Dabei wird der Ausdruck $\sum_i \sqrt{(g_i - h_i)^2}$ minimiert, wobei g_i der Mittelpunkt einer Dreiecksseite auf $\hat{P}$ ist, r_i ist der Schnittpunkt einer Normalen von g_i aus mit einer Dreiecksseite von P, und h_i ist dann die Normale von dieser Dreiecksseite von P geschnitten mit $\hat{P}$.





Nach dem Alignment wird ein resampling durchgeführt: von jedem Vertex von P wird ein Strahl in Richtung Normalvektor ausgesendet, der mit $\hat{P}$ geschnitten wird. Der Normalvektor wird berechnet als Durchnitt über die Normalvektoren aller Dreiecke, die an dem Vertex anliegen. Wird kein Schnittpunkt in einem bestimmten Umfeld gefunden, wird als Default der vertex von P verwendet.

Transformations-basiertes WM für 3D-Meshes IV

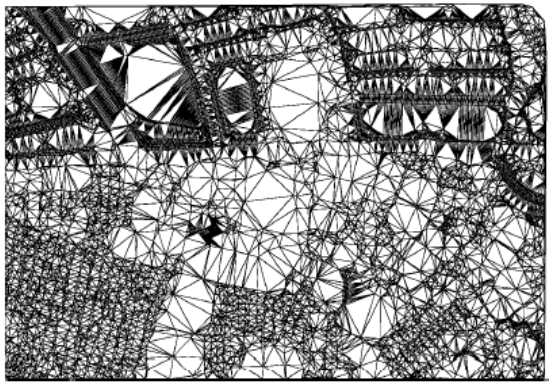


(a) Original model (b) Watermarked with $\beta = 0.001$. (c) Watermarked with $\beta = 0.005$

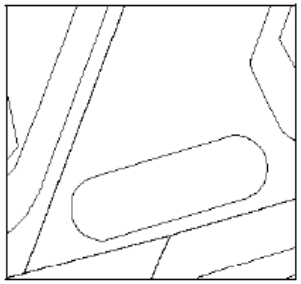
In der Graphik sieht man zunehmende WM Stärke. Ganz rechts sind Artefakte sichtbar, am ehesten vergleichbar mit einer grob knobbeligen Oberfläche.

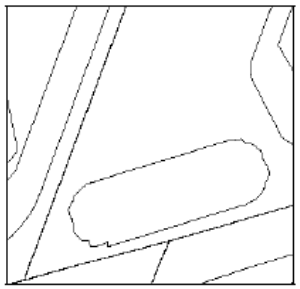
Spektral Transformations-basiertes WM für 3D-Meshes: Anwendung

Eine planare Anwendung ist das WM von Kartenmaterial. Zuerst wird die Digitalisierte Karte einer Delaunay Triangulierung unterzogen um ein einzelnes verbundenes mesh zu erzeugen. Alle Kreuzungspunkte der Karte werden als vertices verwendet.

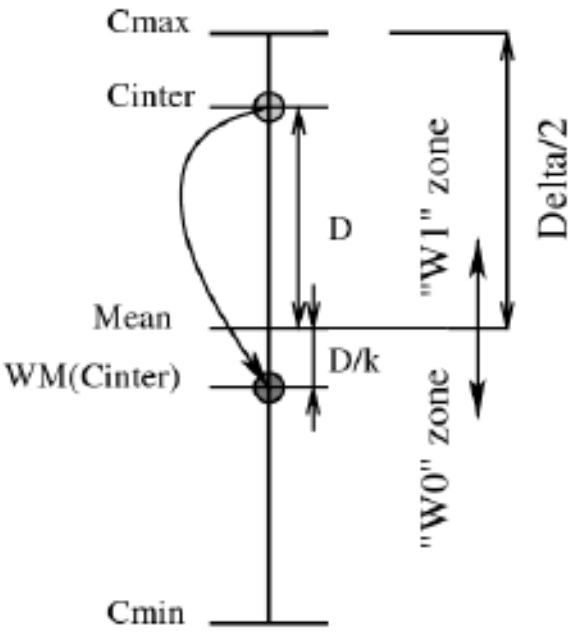


Anschliessend werden Patches zum redundanten Einbetten (und zur Komplexitätsreduktion) generiert in Abhängigkeit von der Anzahl der lokalen vertices. Dann wird das WM eingebettet und die ursprüngliche Kartenkonnektivität wiederhergestellt.





Blind Spektral Transformations-basiertes WM für 3D-Meshes



Das mesh wird einer Spektraltransformation unterzogen und die Koeffizienten $r_i = (r_{i,x}, r_{i,y}, r_{i,z})$ werden extrahiert. In jedes dieser Koeffizienten Tripel i kann nun durch quantisierungsbasiertes WM ein bit eingebettet werden, zur Erhöhung der Robustheit werden nur die Triplets i der niederen und mittleren "Frequenzen" verwendet.

Nach der Bestimmung von $a = \max(r_{i,x}, r_{i,y}, r_{i,z})$ und $b = \min(r_{i,x}, r_{i,y}, r_{i,z})$ wird $mean = \frac{a+b}{2}$ berechnet, der zur Unterschiedung für das eingebettete WM bit dient: wird eine 0 eingebettet, muss der "mittlere" Koeffizient in $[b, mean]$ liegen, ansonsten in $[mean, a]$.

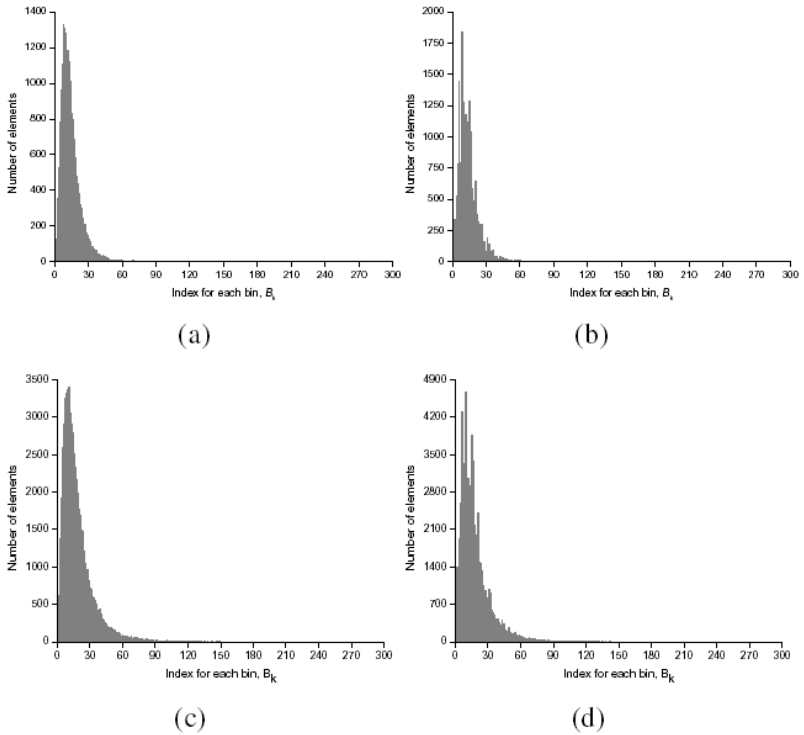
Der Abstand zwischen dem $mean$ und dem eventuell veränderten Koeffizienten bestimmt die Robustheit (die generell wie bei den meisten quantisierungsbasierten Verfahren auch hier eher gering ist). Im Schnitt muss nur jeder zweite Koeffizient verändert (quantisiert) werden.

Mesh WM von Laplace Koordinaten Histogrammen I

Nach der Berechnung der Kirchhoff Matrix K werden für jeden vertex $v_i = (x_i, y_i, z_i)$, $1 \leq i \leq n$ die Laplace Koordinaten $L_i = (x'_i, y'_i, z'_i)$ berechnet durch $L = K \times O$ wobei in den n Zeilen von O die n vertices mit ihren kartesischen Koordinaten stehen. Für jeden vertex wird $d_i = \sqrt{(x'^2_i + y'^2_i + z'^2_i)}$ bestimmt, alle d_i werden in Histogramm-bins eingeteilt. Diese Histogramme bzw. Paare deren Bins sind die WM primitives.

Das bipolare WM $\in \{-1, 1\}$ wird in ein "gültiges" Paar von Bins (B_{k1}, B_{k2}) eingebettet (benachbarte Bins werden verwendet), d.h. die Gesamtanzahl der Elemente in beiden Bins ist ausreichend gross, und das erste und letzte Bin wird nie verwendet: $|\hat{B}_{k1}| < |\hat{B}_{k2}|$ für das Einbetten eines -1, $\geq$ für 1. D.h., dass die Belegung der Histogramm-bins verändert wird. Um das zu erreichen, werden einzelne d_i so verändert, dass sie ihre Bin-Zugehörigkeit entsprechend ändern. Nur wenn die obige Bin-Relation nicht erfüllt ist, muss überhaupt manipuliert werden.

Mesh WM von Laplace Koordinaten Histogrammen II



Für das WM Einbetten wird die notwendige Anzahl von kleinsten Elementen von B_{k+1} nach B_k "verschoben" durch $\hat{d}_i = d_{mean}$, wobei d_{mean} der Durchschnitt der d_i's von B_k ist. Anschliessend werden entsprechend manipulierte Laplace Koordinaten $\hat{L}_i$ berechnet mit $||\hat{L}_i - L_i||^2 = (\hat{x}'_i - x'_i)^2 + (\hat{y}'_i - y'_i)^2 + (\hat{z}'_i - z'_i)^2$ sodass $\hat{x}'^2_i + \hat{y}'^2_i + \hat{z}'^2_i = \hat{d}^2_i$.

Es stellt sich heraus, dass $\hat{x}'_i = \frac{x'_i \hat{d}_i}{\sqrt{(x'^2_i + y'^2_i + z'^2_i)}}$, analog für $\hat{y}'_i, \hat{z}'_i$. Abschliessend werden die kartesischen Koordinaten des mesh berechnet durch $K^{-1} \times \hat{L}$. Auslesen geht blind und analog zur Einbettung.

Mesh WM in sphärischen Koordinaten I

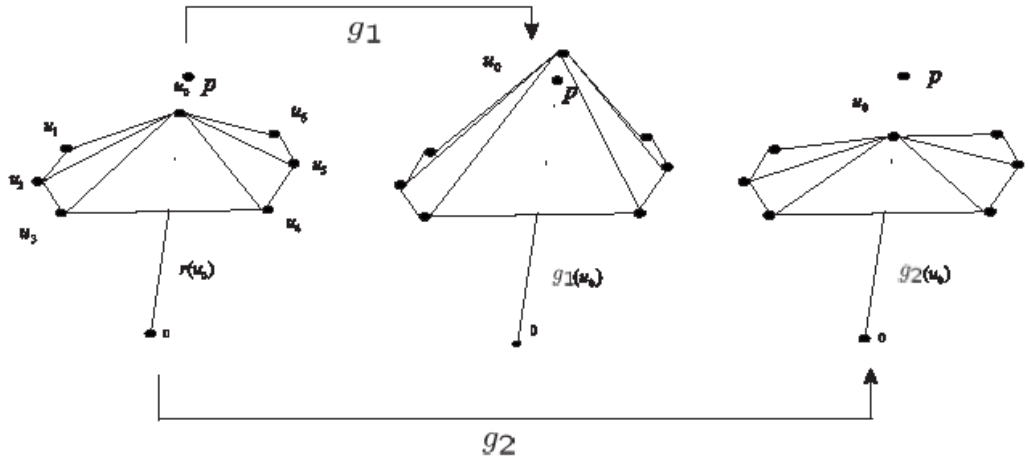
Für jeden vertex $v_i = (x_i, y_i, z_i)$ werden die sphärischen Koordinaten $v_i^s = (r_i, \Phi_i, \phi_i)$ berechnet wie folgt:

- Translation des Schwerpunkts in den Koordinatenursprung.
- Alignment der Hauptachse: Das 3D mesh wird so rotiert sodass seine PCA Hauptachse der vertices mit der z-Achse übereinstimmt. Die PCA Hauptachse entspricht dem Eigenvektor der dem grössten Eigenwert der Kovarianzmatrix der vertex Koordinaten entspricht.
- Konvertierung zu sphärischen Koordinaten (seien der Einfachheit halber (x_i, y_i, z_i) die bereits rotierten und verschobenen Koordinaten): $r_i = \sqrt{(x_i^2 + y_i^2 + z_i^2)}$, $\Phi_i = \arccos(z_i/r_i)$, $\phi_i = \arctan(y_i/x_i)$, mit entsprechenden Wertebereichen.

Der Einbettungsvorgang verändert nur die r_i Komponente, die zu markierenden vertices werden ausgewählt mit einem Zufallszahlengenerator, der einen Winkel $\hat{\Phi}_i$ generiert: der vertex v_i^s dessen Φ_i am nächsten liegt wird als nächster gewatermarked.

Mesh WM in sphärischen Koordinaten II

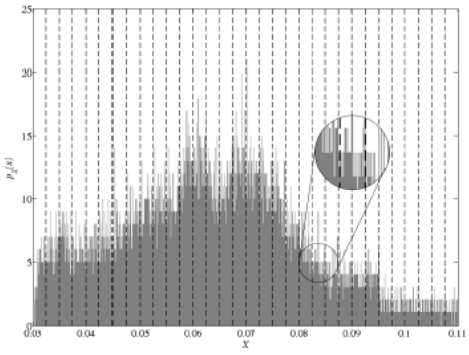
Das Einbetten der WM Information geschieht nun wie in der Graphik dargestellt. Für WM +1 wird eine Funktion g_1 auf den gewählten vertex P angewendet, indem r_i auf einen Wert gesetzt wird der dem α-fachen des Durchschnitts- oder Medianwerts der r_i der direkten Nachbarn entspricht, für WM -1 wird g_2 entsprechend definiert.

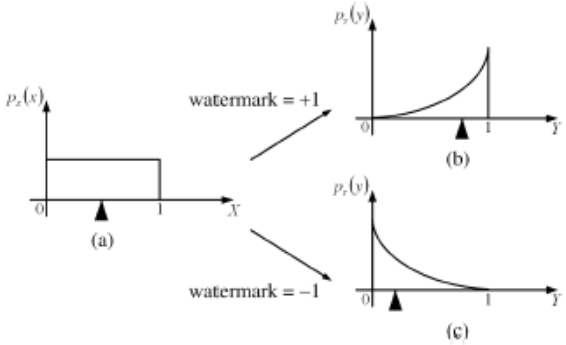


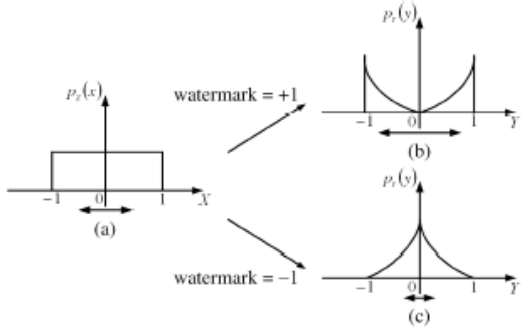
Vertices die für die Bestimmung der Modifikation eines anderen vertex verwendet wurden, werden nicht mehr markiert. Der Ausleseprozess ist anaolg und vergleicht für jeden gewählten vertex r_i mit dem Durchschnitts- oder Medianwerts der r_i der direkten Nachbarn, das Vorzeichen der Differenz ergibt das entsprechedne WM Bit. Jedoch ist dieses WM Verfahren nicht robust gegen remeshing und mesh simplification.

Mesh WM in sphärischen Koordinaten III

Um die Robustheit gegenüber diesen Angriffen zu erreichen, werden von verschiedenen Autoren Histogramme der r_i verwendet um WM einzubetten. Dafür werden n Bins für n einzubettende WM Bits erzeugt und die r_i in jedem Bin normiert. In jedes Bin (für das uniform Distribution der r_i angenommen wird) wird ein WM Bit eingebettet durch Anhebung / Absenkung des Mittelwerts (Methode I) oder der Varianz (Methode II) im Bin.

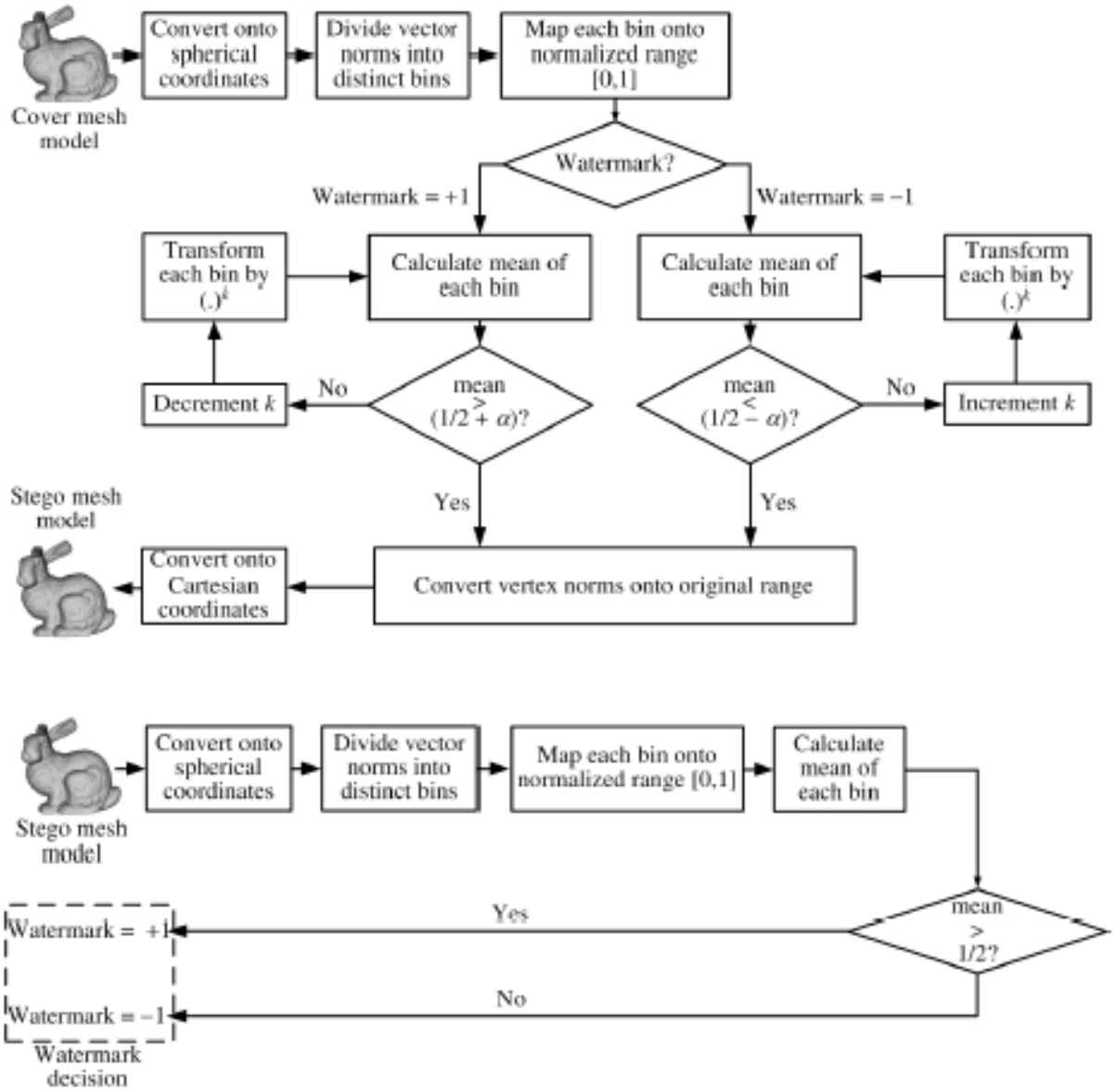




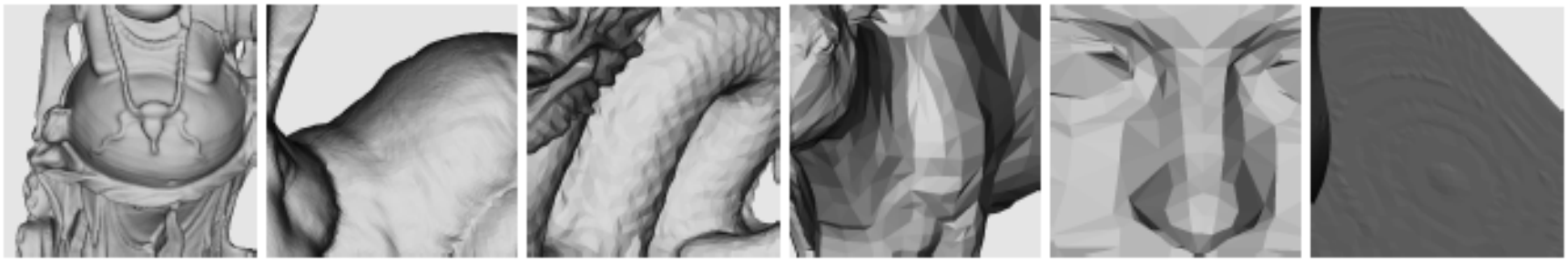


Die Manipulation der r_i geschieht durch Transformationen ähnlich denen verwendet in der Histogrammmodifikation für Grauwertbilder. Anschliessend werden die kartesischen vertex Koordinaten rekonstruiert. Zum Auslesen der WM Information werden die MW (die Varianz) im Bin mit dem erwarteten Wert (z.B. $1/2$ für den MW für normierte r_i) verglichen und das Bit entsprechend ausgelesen.

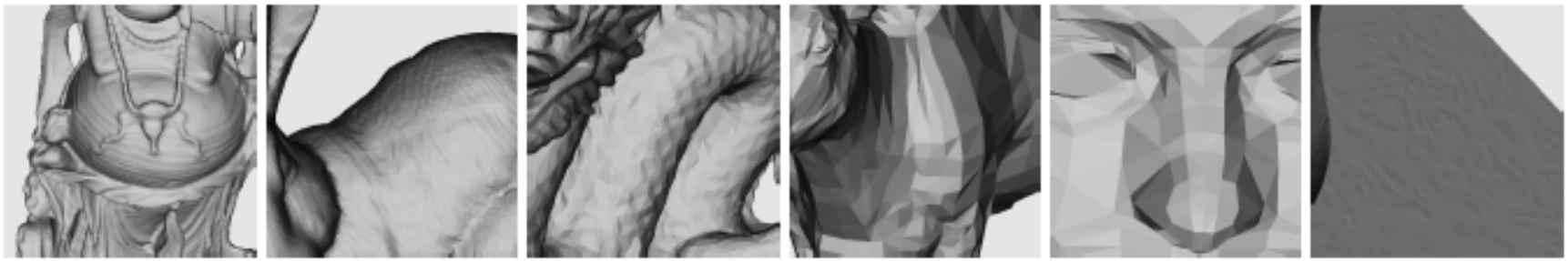
Mesh WM in sphärischen Koordinaten IV



Mesh WM in sphärischen Koordinaten V: Beispiele



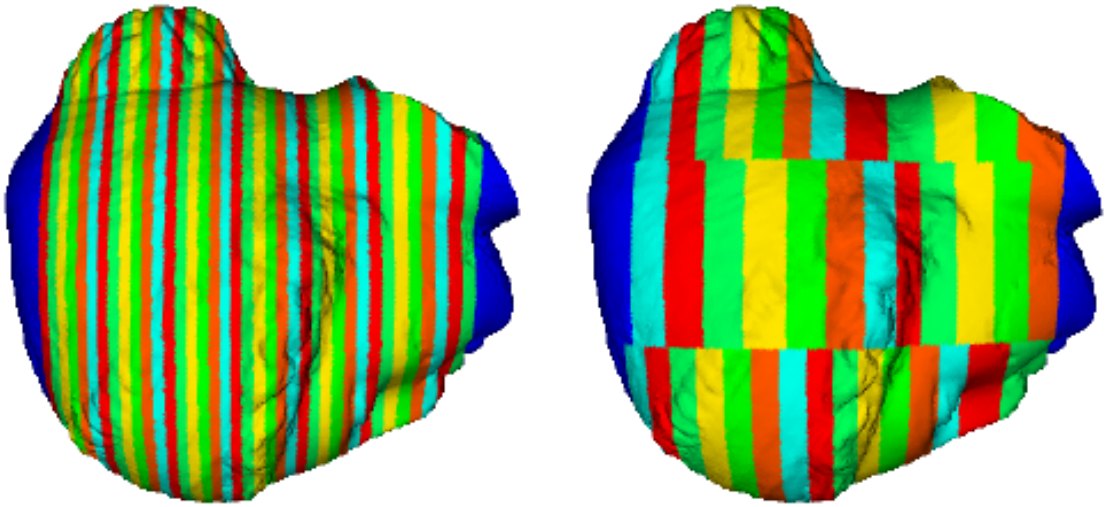
(a) (b) (c) (d) (e) (f)



(g) (h) (i) (j) (k) (l)

Spektral Transformations-basiertes WM für 3D-Meshes: High Frequency I

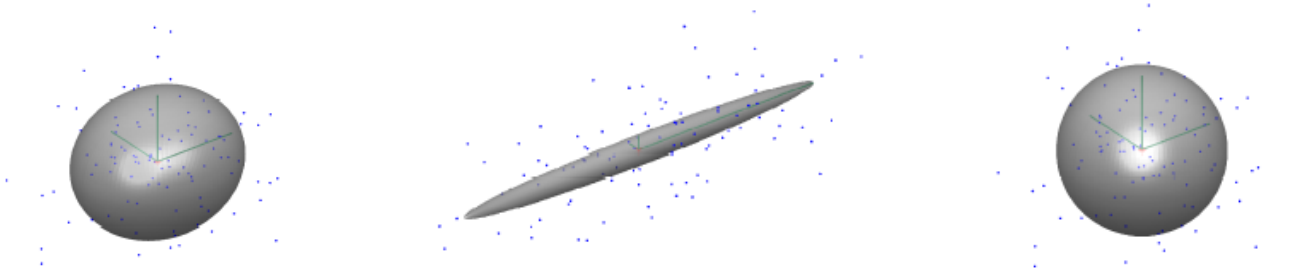
Die früher besprochenen auf der spektral-Transformation beruhenden Methoden betten in den mittleren oder niederfrequenten Koeffizienten ein. Alternativ gibt es den Vorschlag in die 70% der höchsten Koeffizienten einzubetten, allerdings nicht in einzelne Koeffizienten, sondern in deren statistische Verteilung. Nach einem robusten alignment wird das mesh in kleiner Teile aufgeteilt, in die jeweils ein einzelnes Bit eingebettet wird.



Auf die Patches wird die Spektral Transormation angewendet und die hohen Frequenzen im Folgenden betrachtet.

Spektral Transformations-basiertes WM für 3D-Meshes: HF II

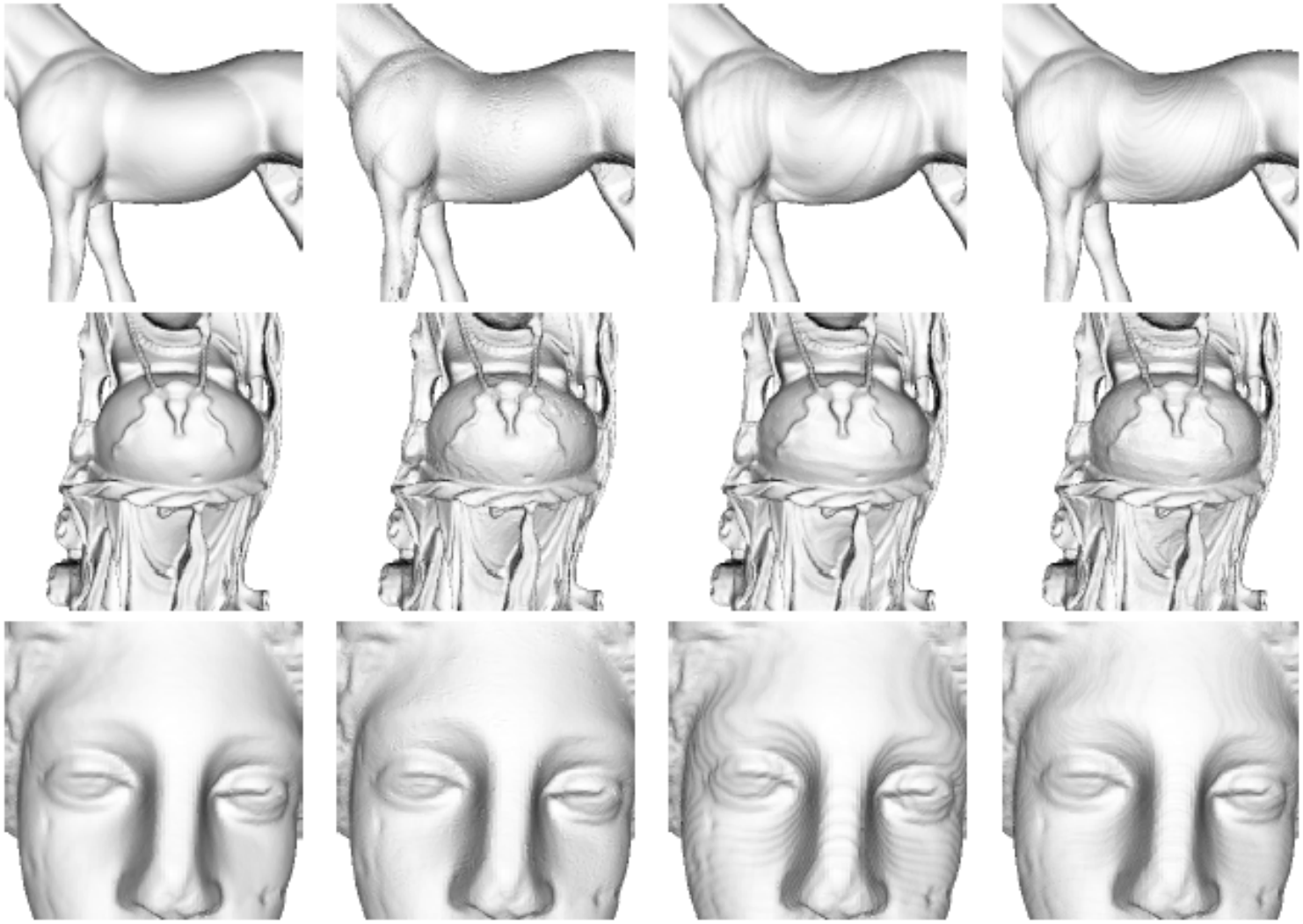
Die Punktwolke der hochfrequenten Koeffizienten wird mit PCA analysiert, die drei Eigenwerte geben die Ausdehnung (Varianz) der Punktwolke in die Richtung der drei Achsen die von den Eigenvektoren aufgespannt werden an. Zuerst wird die Punktwolke so rotiert, dass die Achsen den orthogonalen Eigenvektorrichtungen entspricht.



(b) Ellipse of the original coefficients (c) Ellipse with 1 embedded (d) Ellipse with 0 embedded

Für die Einbettung wird die Punktwolke in Richtung einer Ellipse (WM ist 1) oder eines Kreises (WM ist 0) deformiert, was über eine Modifikation der Varianz (Eigenwert) bzgl. der zweiten und dritten Achse geschieht, was die Varianz bzgl. der ersten Achse unberührt lässt (d.h. dass die Koeffizientenvektoren entsprechend manipuliert werden). Nach der Rückrotation (Rekonstruktion der Spektralkoeffizienten) werden die kartesischen Koordinaten rkonstruiert. Auslesen analog, blindes Verfahren, allerdings sehr sensitiv geg. Alignment- und Patchgenerierungsproblemen.

Spektral Transformations-basiertes WM für 3D-Meshes: HF III



(a) Original (b) Spectral (c) ChoMean (d) ChoVar

Steganographie I

Auch in der Steganographie werden wie beim Watermarking Daten in ein Covermedium eingebettet. Durch die verschiedenen Zielapplikationen gibt es jedoch wichtige Unterschiede:

- Beim Watermarking hat die eingebettete Information immer etwas mit dem eigentlichen Objekt (also dem Covermedium) zu tun
- Ein Angriff gegen ein WM System hat die Entfernung/Zerstörung des WM zum Ziel, ein Angriff gegen ein Steganographisches System hat das Ziel eine eingebettete Nachricht zu entdecken. Daher ist Robustheit gegenüber Manipulationen keine notwendige (wenn auch wünschenswerte) Forderung an ein Stego System.
- Die Datenmengen werden bei Stego Applikationen im Allgemeinen grösser sein.
- Bei Stego Applikationen wird kaum das Original bei der Extraktion zur Verfügung stehen (also immer blind Verfahren).

Seganographie wird von Bürgerrechtsgruppen gern als Antwort auf Einschränkungen bei der Benutzung von starker Kryptographie (USA) und auf das ECHOLON System gesehen.

Steganographie II

Steganographie hat eine lange historische Tradition:

- Technische Steganographie
  - ⋆ Kopfrasur mit Tätowierung und anschliessendem Wiederbewuchs (400 v.Chr.)
  - ⋆ Wachstafeln: Informationen hinter der Wachsschicht versteckt
  - ⋆ Unsichtbare Tinte
  - ⋆ Informationen als kleine Löcher oder Punkte kodiert (z.B. WitnesSoft von Aliroo)
  - ⋆ Microfilm/Microdots als Beistriche in Texten
- Linguistische Steganographie
  - ⋆ Bestimmte Buchstabenpositionen im Cover Text ergeben die versteckte Nachricht (im alten China mit Papiermasken in die Löcher geschnitten waren realisiert).

Steganographie III

Die meisten der verfügbaren Stego Tools (e.g. StegoDos, SysCop, S-Tools, Mandelsteg, EzStego, Hide and Seek, Hide4PGP, White Noise Storm, Steganos) verwenden eine Form von LSB Ersetzung oder Farbpalettenmanipulation. Sie sind damit nicht robust und können mit Verfahren wie *Stirmark*, *Checkmark*, oder *UnZign* sofort entfernt werden. Auch sind sie selbst gegenüber moderater Kompression nicht resistent – WARUM ??

Ein Angreifer geg. ein Stego-System kann *passiv* sein (Nachrichten werden getestet, im Entdeckungsfall blockiert, keine Manipulation), *aktiv* sein (unspezifische Manipulationen da unklar ist welches System verwendet wird) oder aber *bösartig - malicious* (spezifische Angriffe gegen ein bestimmtes Stego-System, falsche Nachrichten einschleusen). Meist wird der passive Angreifer angenommen, da die anderen Szenarien eher im WM Bereich behandelt werden.

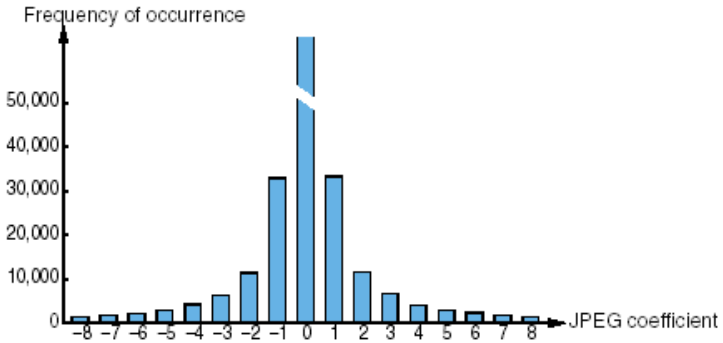
Steganographie – Steganalysis

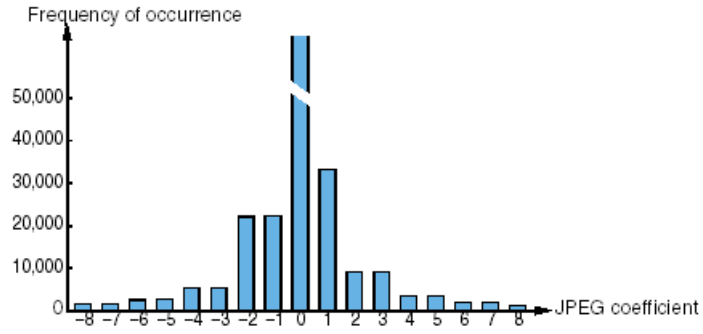
Es gibt drei verschiedene Arten von Steganalysis, die sich (auch) durch das unterschiedliche Wissen das ein Angreifer hat, differenzieren lassen:

- **Targeted Steganalysis**: in diesem Szenario wird angenommen, dass der Angreifer detaillierte Kenntnisse über den verwendeten Einbettungsalgorithmus und dessen Funktionsweise hat (d.h. strukturelle Schwächen können gezielt ausgenutzt werden). Ziel ist hier nicht nur die binäre Antwort sondern nach Möglichkeit auch eine Abschätzung der Länge der eingebetteten Daten.
- **Blind / Universal Steganalysis**: typischerweise werden hier Klassifikationsverfahren mit vorausgehender Merkmalsextraktion verwendet. Klassischerweise werden Trainingsdaten verwendet, um den Classifier auf Stegoworks und Coverworks zu trainieren (2 Klassen). Da der Einbettungsalgorithmus unbekannt ist, müssen für die Generierung der Trainingsdaten möglichst viele verschiedene Stegoalgorithmen verwendet werden (und der verwendete ev. gerade nicht). Hier ist oft der 1-Klassenfall besser (nur Coverworks werden trainiert).
- **Forensic Steganalysis**: Ziel ist hier das Auslesen der Nachricht oder Identifizierung des Einbettungsalgorithmus oder des verwendeten Schlüssels (stegowork-only attack, known cover attack, known-stego method attack, known message a ttack etc.).

Steganographische Algorithmen: LSB I

LSB Einbettung taugt zwar nicht als robustes Wasserzeichen, das ist jedoch für Steganographie unerheblich. Wird eine 0 in 51 eingebettet, $(51)_2 = 00110011$, so wird das letzte bit (LSB) durch 0 ersetzt, $00110010 = (50)_2$. Umgekehrt wird 50 auf 51 abgebildet, soll im LSB eine 1 eingebettet werden. In Analogie kann der gesamte Wertebereich in "Pairs of Values" (PoV) $(2i, 2i+1)$ aufgeteilt werden, die aufeinander abgebildet werden. Durch das LSB einbetten werden im Histogramm solche Paare auf sehr ähnliche Werte abgebildet, da die einzubettenden Daten als zufälling angesehen werden.





Völlig analog kann auch im DCT Bereich LSB Einbetten in quantisierte Koeffizienten gemacht werden (mit der Ausnahme dass viele Koeffizienten 0 oder 1 sind – die ausgelassen werden müssen – was die Kapazität der Verfahren verringert). JSteg ist so ein Verfahren bei dem die Änderung im DCT Koeffizienten Histogramm sehr deutlich sind, die wie im spatial domain Fall mit statistischen Methoden feststellbar sind.

Steganographische Algorithmen: LSB II

Histogrammveränderungen durch LSB Einbetten kommen allerdings nur bei sehr hohen Nachrichtenlängen zum Tragen. Eine Methode die nicht nur LSB Einbettung sondern auch die Länge der Nachricht schätzt heisset “sample pair analysis”. Man betrachtet im Bild die Menge P aller Paare horizontal benachbarter Pixel. Es werden definiert:

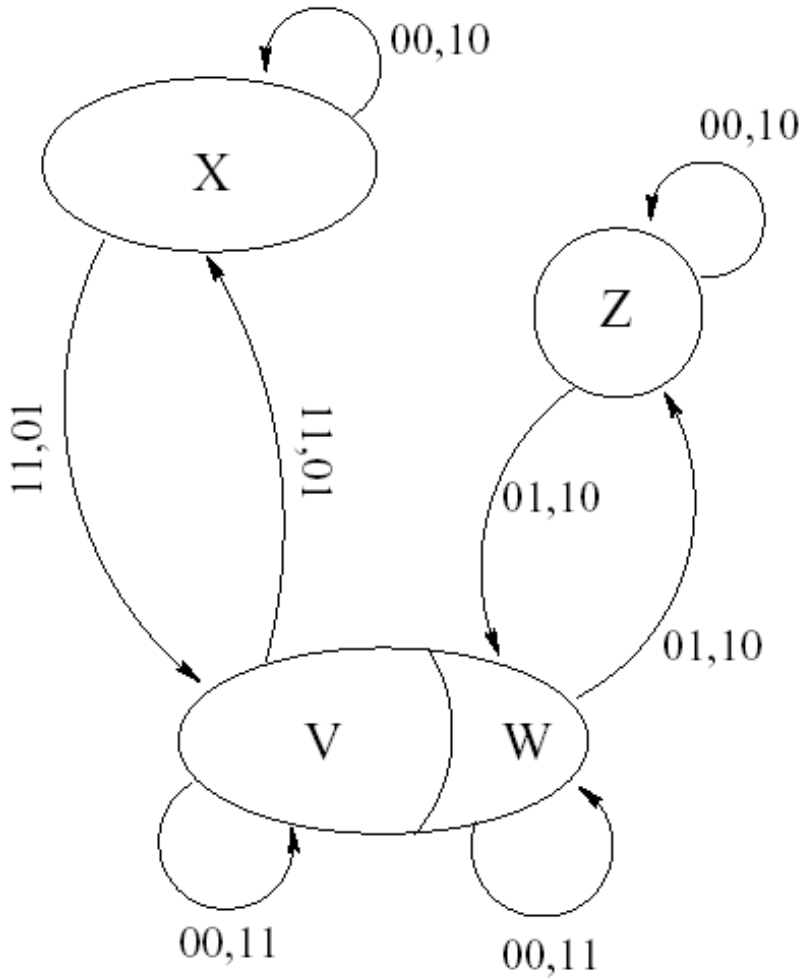
- X als die Menge von Paaren $(u,v) \in P$ sodass v gerade ist und $u < v$ oder v ungerade ist und $u > v$.
- Y als die Menge von Paaren $(u,v) \in P$ sodass v gerade ist und $u > v$ oder v ungerade ist und $u < v$.
- Z als die Menge von Paaren $(u,v) \in P$ sodass $u = v$.

- Weiters wird Y in zwei Teilmengen W und V geteilt, wobei W nur Paare der Form $(2k, 2k+1)$ und $(2k+1, 2k)$ enthält und $V = Y - W$. X und Y werden als gleich gross angenommen (realistisch, Gradienten in beide Richtungen gleich häufig).

Durch LSB Einbettung können nun die folgenden 4 Situationen entstehen (X, V, W, Z ergeben ganz P und heissen primary sets):

00) both values u and v remain unmodified;
01) only v is modified;
10) only u is modified;
11) both u and v are modified.

Steganographische Algorithmen: LSB III



Durch die verschiedenen Übergänge wechseln die Paare die Mengenzugeörigkeiten. Nun werden für die verschiedenen Übergangssituationen die erwartete Menge von modifizierten Pixelpaaren berechnet, ausgedrückt durch die Nachrichtenlänge q. Nun können für alle Mengen ihre Kardinalität nach dem Einbetten ausgedrückt werden (z.B. $|X'|$) als Funktionen der Kardinalität vor dem Einbetten und q. Das Ziel ist q auszudrücken nur durch die Kardinalitäten der Mengen nach dem Einbetten:

$$\frac{\gamma q^2}{2} + (2|X'| - |P|)q + |Y'| - |X'| = 0$$

mit $\gamma = |W'| + |Z'|$ und γ ist bei natürlichen Bildern kaum 0.

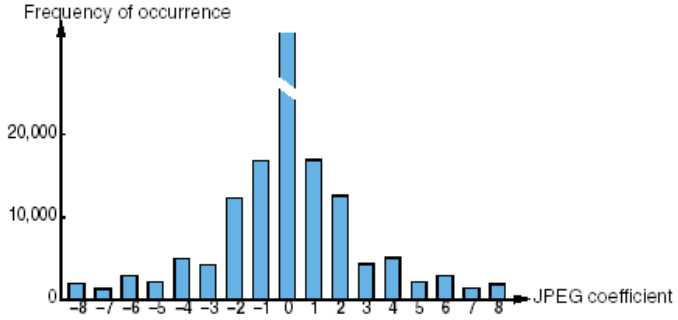
Steganographische Algorithmen: Outguess & F3

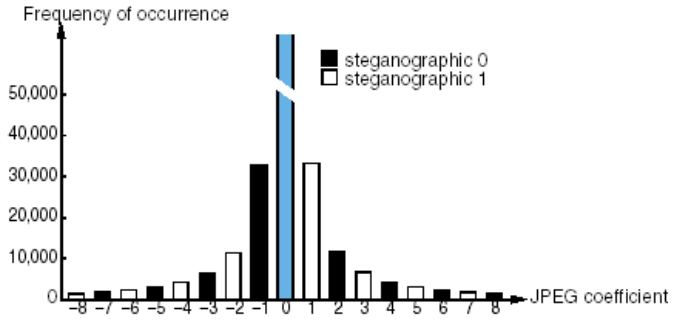
Outguess ist ein LSB Einbettungsverfahren für DCT Koeffizienten. Das Paar $(0, 1)$ wird nicht modifiziert da 0-Koeffizienten durch ihre hohe Anzahl und den visuallen Impact nicht manipuliert werden können und 1-Koeffizienten durch die PoV Struktur nach 0 geändert werden (Balance des Histogramms !). Bei den restlichen Koeffizienten werden nicht alle modifiziert, sondern nur so viele, dass das am meisten unbalanzierte Paar im Histogramm nach dem Einbetten wieder eine Binbefüllung bekommt, die der originalen entspricht. Durch die Zufälligkeit der Auswahl können dann die restlichen Paare ebenso korrigiert werden.

F3 überschreibt keine bits sondern verringert den Absolutwert des Koeffizienten im Falle das LSB passt nicht zu dem einzubettenden Bit. Dadurch wird die PoV Struktur aufgebrochen. 0-Koeffizienten werden nicht benutzt, im Gegensatz zu Outguess 1-Koeffizienten aber schon (höhere Kapazität). Zu beachten ist allerdings, dass die Koeffizienten 1 und -1 auf 0 geschrumpft werden (“shrinkage”).

Steganographische Algorithmen: F3 & F4

Der Empfänger kann unbenutzte 0-Koeffizienten aber nicht von Schrumpf-0 unterscheiden, sodass im Fall von shrinkage das betreffende Bit nochmals eingebettet wird. Das führt zu einer höheren Rate von geraden Koeffizienten, da bei den Koeffizienten 1 und -1 beim Einbetten von 1 nichts verändert wird, beim Einbetten von 0 aber immer shrinkage auftritt, sodass der nächste Koeffizient wieder gerade wird. Eine (unelegante) Methode das auszugleichen ist die Verwendung von Nachrichten, die ein korrigierendes Verältnis von 0/1 aufweisen. F4 löst das eleganter.



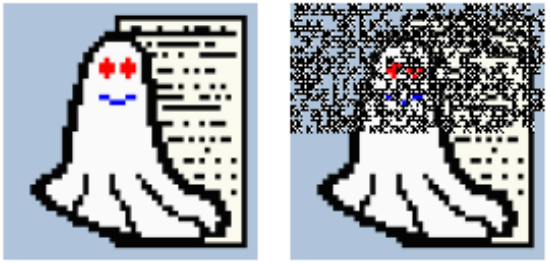


Negative Koeffizienten werden auf inversen steganographischen Wert abgebildet: gerade negative Koeffizienten repräsentieren eine eingebettete 1, ungerade negative Koeffizienten eine 0, bei den Positiven bleibt es wie gehabt. Durch dieses einfache Vorgehen erhält man ein Histogramm mit der erwünschten Form. Es bleibt das Problem der zu vielen 0 Koeffizienten !

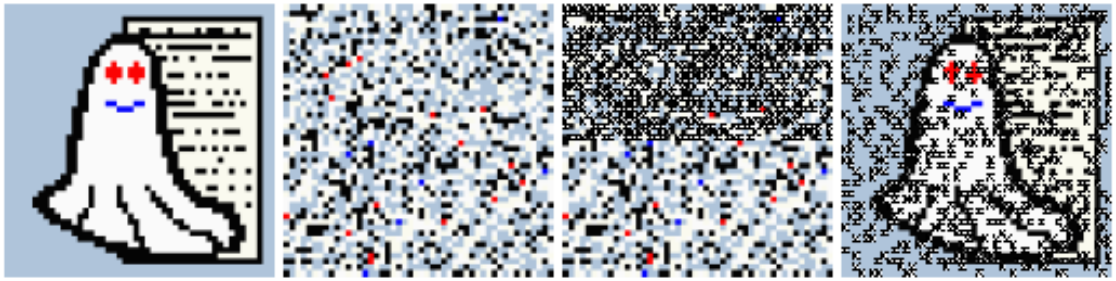
Steganographische Algorithmen: F5 I

F5 nutzt die grundlegende Einbettungstechnik von F4, aller dings mit zwei Verbesserungen: permutatives Einbetten und Matrix Einbetten.

Gewöhnliche sequentielles Einbetten verändert das Covermedium nur bis zu dem Punkt, an dem das letzte Nachrichtenbit eingebettet wird:

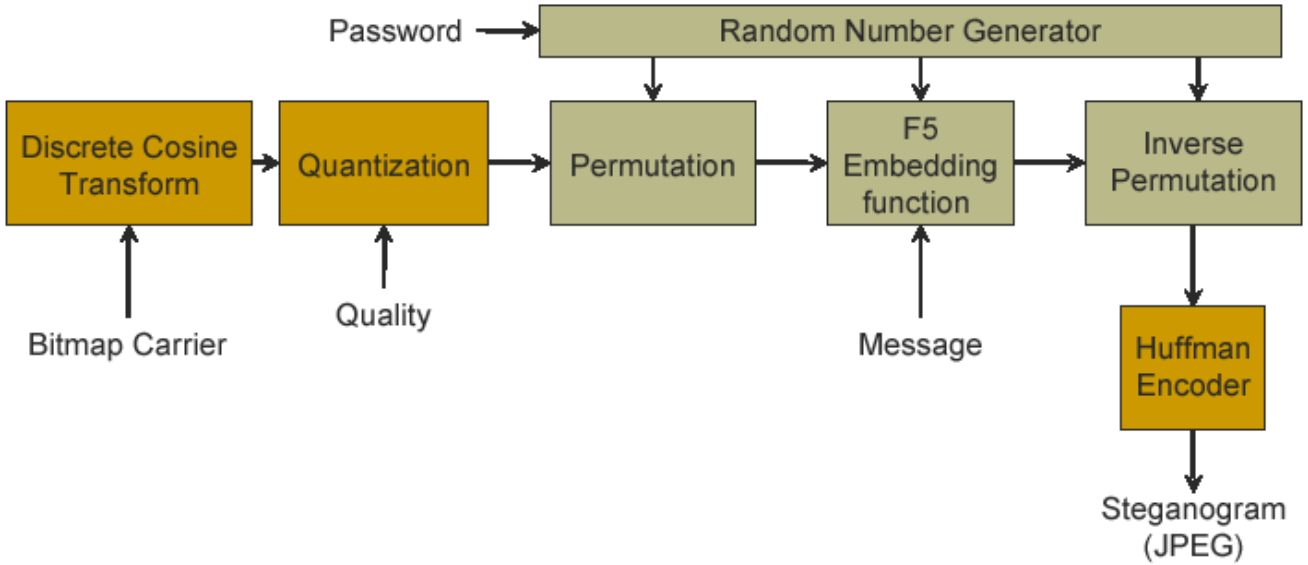


Durch eine vor dem Einbetten vorgenommene Permutation werden die Daten zwar gleich eingebettet, durch die anschliessende Rückpermutationen werden die modifizierten Koeffizienten allerdings gleichmässig über das Bild verteilt.



Steganographische Algorithmen: F5 II

Die Graphik zeigt den wesentlichen Punkt auf, nämlich dass die Rückpermutation nach dem Einbetten aber natürlich vor der inversen DCT erfolgt, was die gleichmässige Verteilung der eingebetteten Daten bewirkt.



Matrix embedding verringert die Anzahl der nötigen bit-Veränderungen per eingebettetem Stego-bit (Einbettungs Effizienz). Typischerweise können 2 bits pro bit-Veränderung eingebettet werden (da in 50% der Fälle keine Veränderung vorgenommen werden muss). Durch shrinkage wird dieser Wert nochmals etwas herabgesetzt, das Matrix Einbetten in F5 kann die Effizienz auf 3.8 bit pro bit-Veränderung gesteigert werden.

Steganographische Algorithmen: F5 III

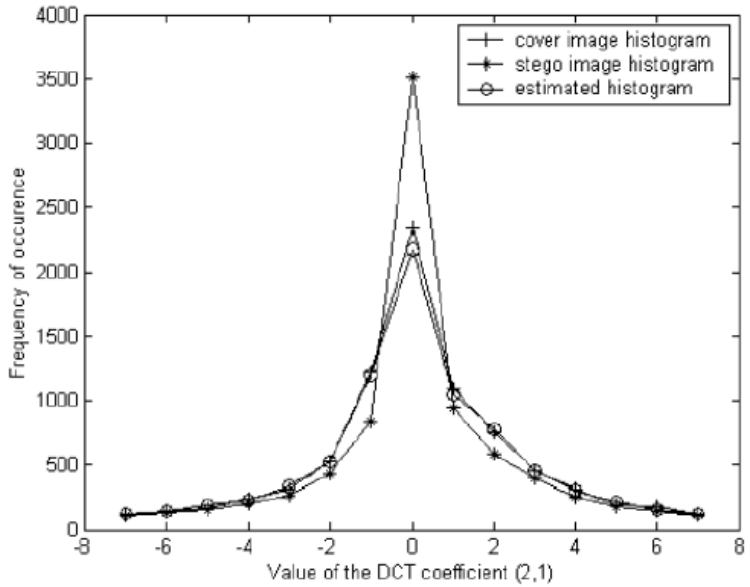
Angenommen, man will zwei bits x_1, x_2 an drei zulässigen Positionen a_1, a_2, a_3 einbetten und dabei maximal eine Position verändern:

$$\begin{aligned}
x_1 = a_1 \oplus a_3,\ x_2 = a_2 \oplus a_3 &\Rightarrow \text{change nothing} \\
x_1 \neq a_1 \oplus a_3,\ x_2 = a_2 \oplus a_3 &\Rightarrow \text{change } a_1 \\
x_1 = a_1 \oplus a_3,\ x_2 \neq a_2 \oplus a_3 &\Rightarrow \text{change } a_2 \\
x_1 \neq a_1 \oplus a_3,\ x_2 \neq a_2 \oplus a_3 &\Rightarrow \text{change } a_3.
\end{aligned}$$

Im allgemeinen Fall hat man ein Codewort a mit n veränderbaren Pitpositionen für k bits der Nachricht x. F ist eine Art von Hashfunktion ide k bits aus einem Codewort "extrahiert", mit Hilfe von Matrix Einbetten findet man ein modifiziertes Codewort a' für alle x und a mit $x = F(a')$ und die Hamming Distanz ist $d(a, a') \leq d_{max}$. D.h., das Codewort mit n bits wird an maximal d_{max} verändert um k bits einzubetten. Im Bsp. ist die Hasfunktion die Verknüpfung der a_i. In F5 ist $d_{max} = 1$ (geht über Hadamard Codewörter).

Steganalysis: Calibration gegen F5

Calibration ist ein Verfahren um Eigenschaften des Coverwork aus dem vermuteten Stegowork abzuleiten. Im konkreten Fall geht es um die Anzahl der 0-Koeffizienten (die durch das in F5 verwendete F4 Einbetten zu hoch sind, shrinkage). Das Stegobild wird dekomprimiert, um 4 Spalten gecropped und mit dem gleichen Quantisierungswert wieder komprimiert. Die Abbildung zeigt wie genau die Histogrammeigenschaften abgeschätzt werden können.



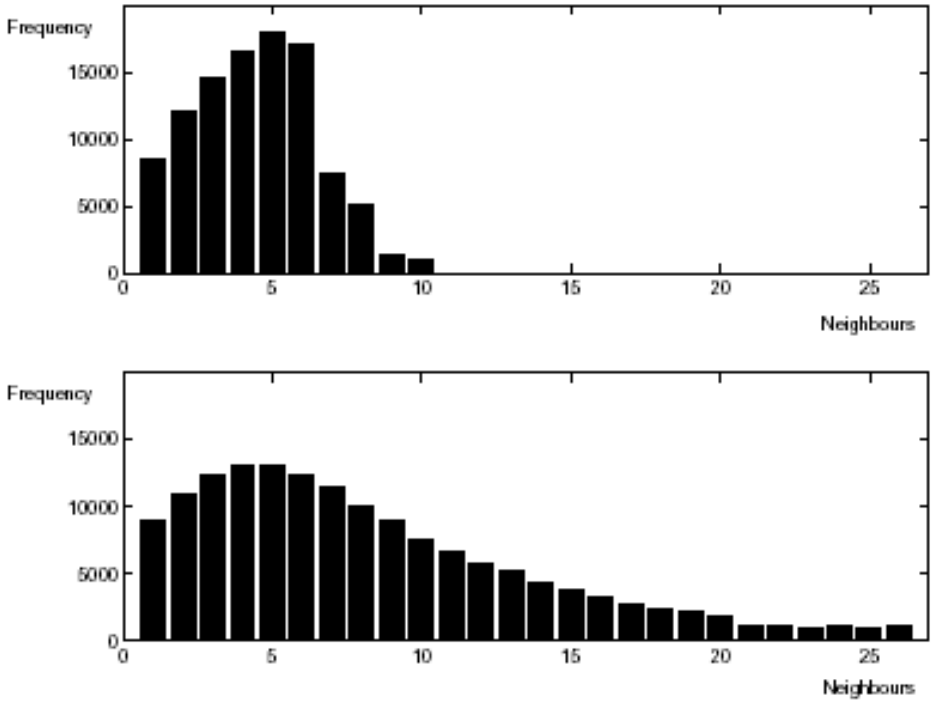
Warum funktioniert das ? Ist der Qualitätsfaktor nicht zu niedrig, ist das Stegobild dem Coverbild visuell und PSNR-objektiv sehr ähnlich. Durch den Shift wird die Block-Struktur der (manipulierten) quantisierten Koeffizienten zerstört und manche Eigenschaften sind dem Coverbild sehr ähnlich. Neben dem Cropping können andere Desynchronisierungsmassnahmen wie Rotation etc. verwendet werden.

Steganalysis: Calibration gegen Outguess

Grundlage ist die Berechnung eines “blockiness” Masses B, das die absoluten Pixeldifferenzen über horizontale und vertikale Grenzen zwischen benachbarten Pixelblöcken berechnet und aufsummiert. B steigt monoton mit der Anzahl der geflippten DCT LSB. Die erste Ableitung (Steigung - Slope) dieses Zusammenhangs ist maximal für das Coverbild und nimmt ab für ein Bild das bereits eine Nachricht enthält.

Zuerst wird $B_S(0)$ bestimmt, d.h. B des dekomprimierten Stegobildes. Dann wird mit Outguess eine Nachricht mit maximaler Länge eingebettet und B des Ergebnisses bestimmt: $B_S(1)$. Nun wird Calibration und Rekompression mit der identischen Quantisierung angewendet und B bestimmt: $B(0)$. Wieder wird eine maximale Nachricht eingebettet und B berechnet: $B(1)$. Und nochmal: $B1(1)$. Nun wird berechnet $S_0 = B(1) - B(0)$ als Slope des Originalbilds ($p = 0$), $S_1 = B1(1) - B(1)$ als Slope eines Bildes mit maximaler Nachricht eingebettet und $S = B_S(1) - B_S(0)$ ist zwischen S_0 und S_1. Die Länge der Nachricht ist $p = \frac{S_0 - S}{S_0 - S_1}$ (durch lineare Interpolation aus $S = S_0 - p(S_0 - S_1)$). Ist $p = 0$, ist keine Nachricht eingebettet.

Color Embedding / Steganalysis



Das Stegoverfahren “Hide” bettet in jeweils einen von drei Farbkanäle ein, indem der Absolutwert um 1 erhöht oder erniedrigt wird (anstelle von LSB flipping).
Als Ergebnis sieht man dass es eine deutlich höhere Anzahl von “benachbarten” Farben pro Farbpixeln gibt, als bei natürlichen Bildern vorkommen (z.B. 26 Nachbarn statt 10; Nachbarn sind solche Farben die sich in nur einem Farbkanal um einen Wert unterscheiden). Das kann statistisch erkannt werden.

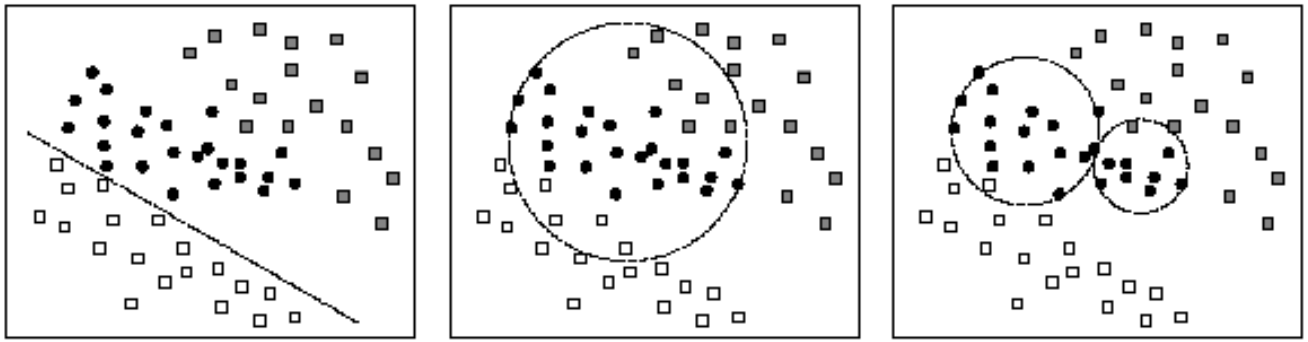
Blind / Universal Steganalysis I

Der Effekt von steganographischen Methoden im Bildbereich ist eine Hinzufügung von Rauschen, das Histogramm wird glatter – das kann als eine Tiefpassfilterung des Histogramms interpretiert werden. Diese Beobachtung kann benutzt werden indem das Histogramm einer Fouriertransformation ("histogram characteristic function") unterzogen wird: die höheren Frequenzen haben deutlich weniger Energie verglichen mit Coverimages.

Idealerweise möchte man ein Verfahren verwenden ähnlich der calibration, um das Coverwork schätzen zu können. Da spatial domain Algorithmen häufig als Addition von hochfrequentem Rauschen modelliert werden, dienen dazu Denoising Algorithmen. Beispielweise werden die ersten k zentralen Momente $\sum_{i,j} |r[i,j] - mean(r)|^k$ von Rausch-Residuals r (originaler Wert minus entrauschtem Wert) von Wavelet Transformationskoeffizienten als Features vorgeschlagen.

Blind / Universal Steganalysis II

Ein ähnliches Konzept wird verfolgt von Farid et al []: nach einer Wavelet Zerlegung werden von allen Subbands mean, variance, skew und kurtosis berechnet. Um auch die Abhängigkeiten zwischen Subbands modellieren zu können, werden zusätzlich die analogen Werte für die Fehler eines linearen Prediktors für ein Subband berechnet: der Prediktor ist ein gewichtetes Mittel von benachbarten Koeffizienten der Subbands anderer Farbe, anderer Orientierung oder eines benachbarten Scales, wobei die Gewichte in einem Fehlerminimierungsprozess berechnet werden.



In späteren Arbeiten wird ein one-class SVM verwendet, der nur Coverimages trainiert. Dadurch wird vermieden, dass Trainingsdaten bzgl. Stegoimages grundsätzlich unvollständig sind, da nie alle steganographischen Verfahren bekannt sind.

Blind / Universal Steganalysis III

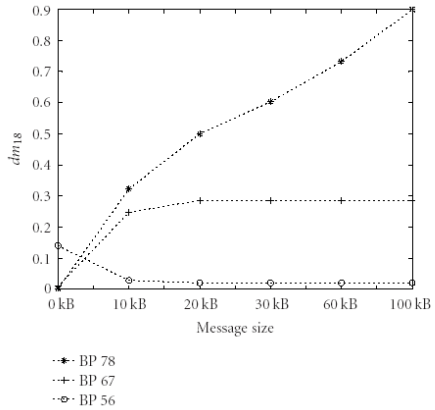
Bild Qualitätsmasse (IQM) wurden auch vorgeschlagen um Stegoimages von Coverimages zu unterscheiden []: wiederum basierend auf einer Schätzung des Coverimages aus dem Stegoimage durch Denoising oder Wiener Filterung werden aus dem Coverimage, dem gefiltertem Coverimage, dem Stegoimage und dem gefiltertem Stegoimage verschiedenste IQM berechnet. Es wurden verschiedene Szenarian betrachtet (aktiver und passiver Angreifer, also robuste WM und steganographische Verfahren).

Durch umfangreiche statistische Analyse stellt sich heraus, dass für die unterschiedlichen Szenarien unterschiedliche IQM am besten geeignet sind, eingebettete Daten zu erkennen. Für jedes Szenario wird ein Set von diskriminativen IQM definiert und deren Werte für (gefilterte) Stegoimages und (gefilterte) Coverimages durch Regression über ein Trainingsdatenset gefittet. Im Fall von Testdaten wird diejenige Klasse gewählt, bei der der geringere Abstand zum Regressionsergebnis auftritt.

Blind / Universal Steganalysis IV

Binäre Ähnlichkeitsmasse werden verwendet, um die vermutete Abnahme der Korrelation zwischen 7. und 8. Bitplane im Falle der LSB Einbettung auszudecken. Als Beispiel werden local binary patterns (LBP) betrachtet, wo für jede Bitplane ein 512-bin Histogramm erzeugt wird basierend auf gewichteten Nachbarschaften, der Wert an einer Pixelposition ist $S = \sum_{i=0}^{7} x_i 2^i$ wobei die Gewichte 2^i den 8 Nachbarn zugeordnet werden wie im Bild. Man bezeichnet als S_n^x die Einträge im n-ten Bin der x-ten Bitplane. Als Beispiel für ein Mass dient die Ojala Kullback-Leibler Distanz: $-\sum_{n=0}^{511} S_n^7 \log \frac{S_n^7}{S_n^8}$.

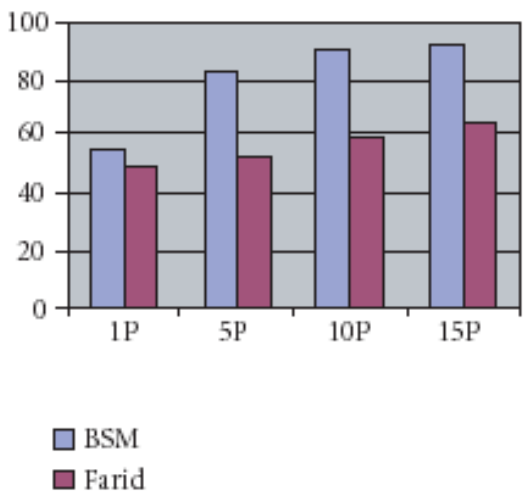
1	2	4
128	256	8
64	32	16

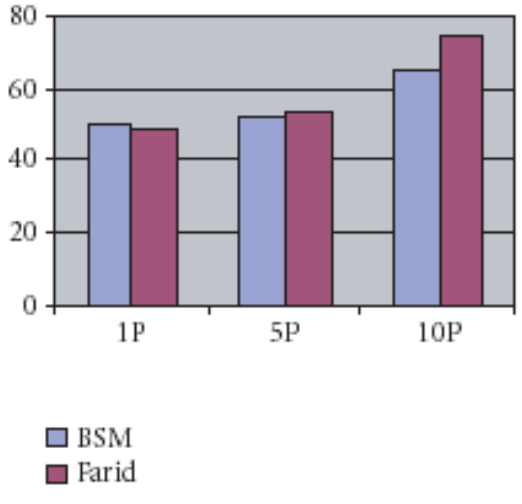


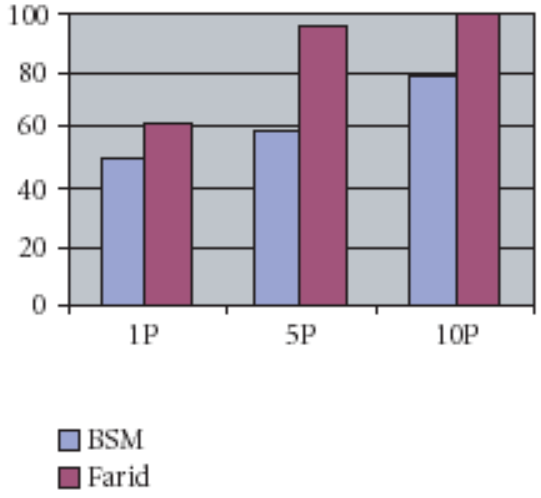
Die Graphik zeigt die Veränderung dieses Masses mit zunehmender Nachrichtenlänge unter LSB Einbettung, zwischen anderen Bitplanes eine keine solche Veränderung messbar. Diese Methodik kann natürlich auf beliebige Bitplanepaare angewendet werden.

Blind / Universal Steganalysis IV: Ergebnisse

Es ist klar ersichtlich, dass für unterschiedliche Einbettungsverfahren verschiedene features gut zur Klassifikation geeignet sind. Dieses Resultat lässt die Verfahren schon viel weniger “universal” erscheinen.

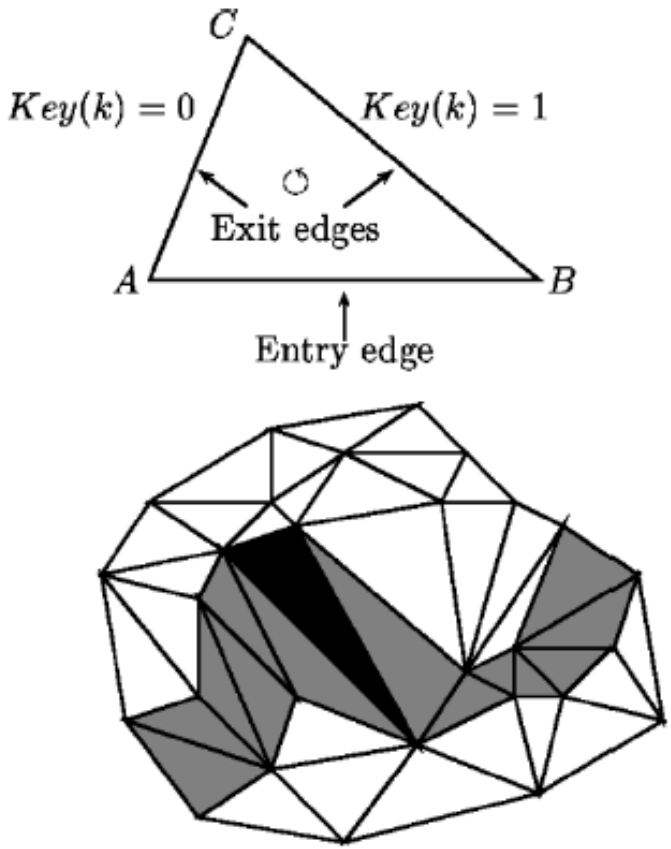






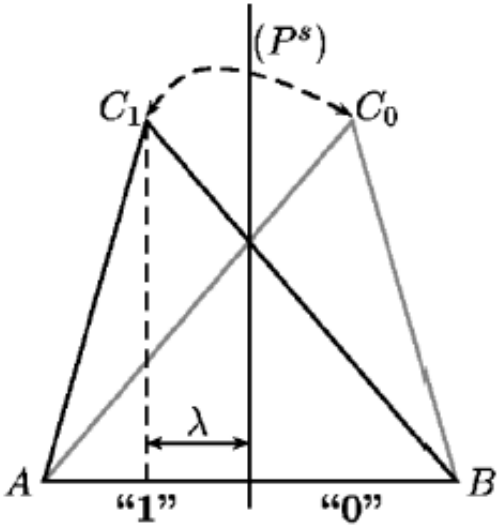
Im Fall der binären Ähnlichkeitsmasse ist eine grosse Menge von verschiedenen Formeln verfügbar, wobei für ein gegebenes Datenset oder Methode erst die passenden gefunden werden müssen. Das lässt die methode von Farid als “more universal” erscheinen.

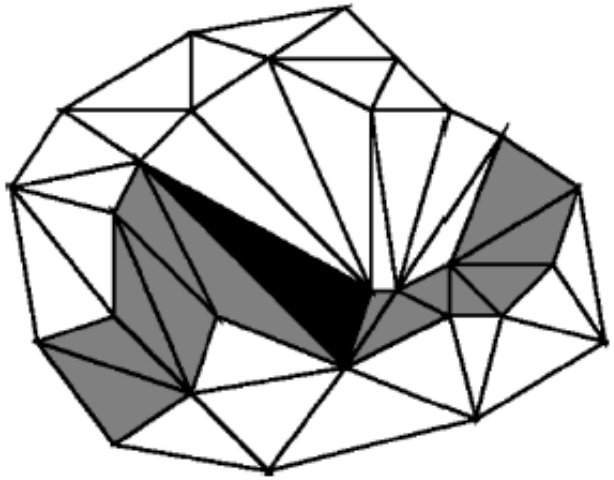
Steganographie für Meshes I



Das vorgeschlagene Verfahren [] benötigt zwei Schritte: erstens eine Liste von Dreiecken die die Nachricht enthalten (deren Auswahl von einem Zufallszahlengenerator gesteuert wird), und zweitens die Modifikation der Dreiecke in der Liste entsprechend dem einzubettenden Symbol. Nach der Identifikation eines Startdreiecks basierend auf speziellen geometrischen Eigenschaften, wird das zweite Dreieck nach dem Wert des Zufallszahlengenerators bestimmt, die exit edge des aktuellen Dreiecks ist dann die entry edge des nächsten Dreiecks. In jedes Dreieck wird ein Bit eingebettet.

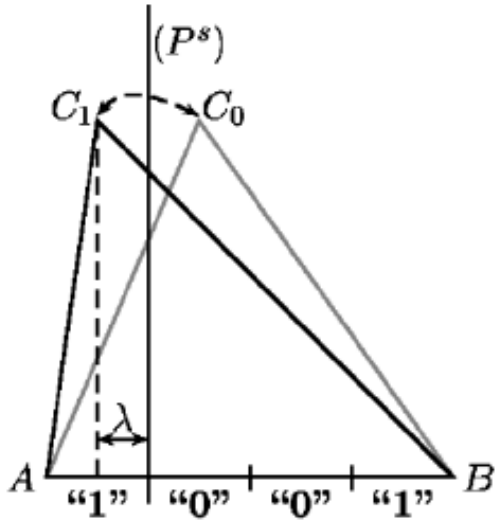
Steganographie für Meshes II

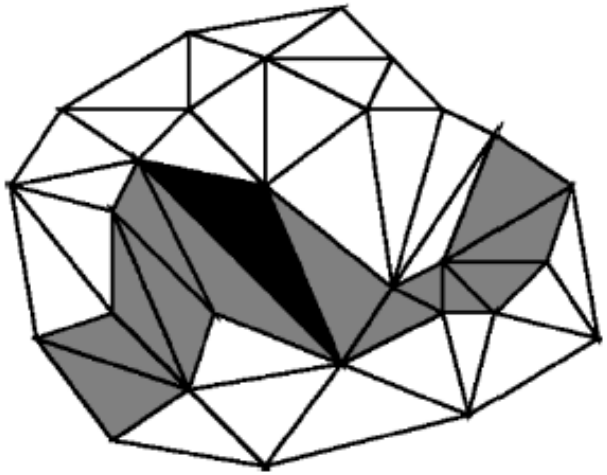




Jedes Dreieck hat zwei Zustände. Der Zustand des Dreiecks ist definiert durch die Position der orthogonalen Projektion der Dreieckspitze über der entry endge auf die entry edge.
Die entry edge ist in zwei Bereiche unterteilt, die den einzubettenden Bits des WM entsprechen. Ist die Projektion bereits im korrekten Bereich ist keine Manipulation notwendig, im anderen Fall wird die Dreieckspitze in ensprechenden bereich verschoben, der Abstand von der Trennline λ legt die Robustheit des Verfahrens, aber auch die Distortion fest.

Steganographie für Meshes III





Im Beispiel sieht man wie die entry edge in vier Partitionen geteilt werden kann, mit den offensichtlichen Folgen: geringere Robustheit aber auch geringerer Impact auf die Wahrnehmbarkeit.
Interessant ist hier die Möglichkeit der Umkehrbarkeit: wird λ und der sog. "erasing key" gespeichert (d.h. die Information, welche Dreiecke modifiziert wurden und welche nicht), kann mit Hilfe der Einbettungszufallszahlenfolge der Einbettungsvorgang umgekehrt werden.

Integritätsprüfung für Multimediale Daten

Bildverarbeitungsalgorithmen und Bildbearbeitungstools ermöglichen die Manipulation von visuellen Daten ohne beweisbare Spuren. Aus diesem Grund sind Verfahren zur Integritätsprüfung solcher Daten notwendig. In der klassischen Kryptographie werden für Applikationen dieser Art Kombinationen von Hash-Funktionen (wie MD-5 oder SHA-1) und Verschlüsselungsverfahren oder digitale Signaturverfahren (wie DSA) verwendet. Bestimmte Eigenschaften multimedialer Daten machen jedoch spezielle bzw. angepasste Verfahren notwendig.

Bei Verfahren der klassischen Kryptographie werden minimale Unterschiede angezeigt, da sie von jedem einzelnen Inputbit abhängen (müssen). Im Bereich der visuellen Daten gibt es viele Manipulationen, die zwar die binäre Repräsentierung der Daten ändern, jedoch keinen oder nur wenig Einfluß auf den wahrnehmbaren visuellen Inhalt haben (z.B. Formatkonvertierung, Kompression, Filterung, ...). Daher wurden Methoden entwickelt, die zwar nicht die Integrität der binären Repräsentation wohl aber die Integrität der visuellen Inhalte sicherstellen sollen: semi-fragile watermarks und robuste multimedia Hash Funktionen.

Integritätsprüfung: Watermarks & Hash Funktionen

Authentication Watermarks dienen zur Überprüfung der Authentizität von visuellen Daten. Es gibt hier “fragile” Wasserzeichen (die ähnlich wie klassische Hash Funktionen kleinste Veränderungen in Bildmaterial anzeigen) und “semi-fragile” Wasserzeichen die nur wahrnehmungsrelevante Bildmodifikationen anzeigen sollen. Die Beschränkung auf wahrnehmeungsrelevante Modifikationen gilt alnalog für robuste Hash Funktionen (auch als “perceptual hashes” bezeichnet).

Hauptvorteil von Wasserzeichen im Gegensatz zu Hashfunktionen ist dass sie integraler Bestandteil des Bildes sind und dass entdeckte Modifikationen meist lokalisiert werden können. Das Bildmaterial wird durch die Einbettung natürlich verändert, was je nach Art des Verfahren auch reversibel sein kann. Hauptvorteil von Hashfunktionen ist, dass das Bildmaterial nicht verändert wird, jedoch können entdeckte Modifikationen nicht/selten lokalisiert werden.

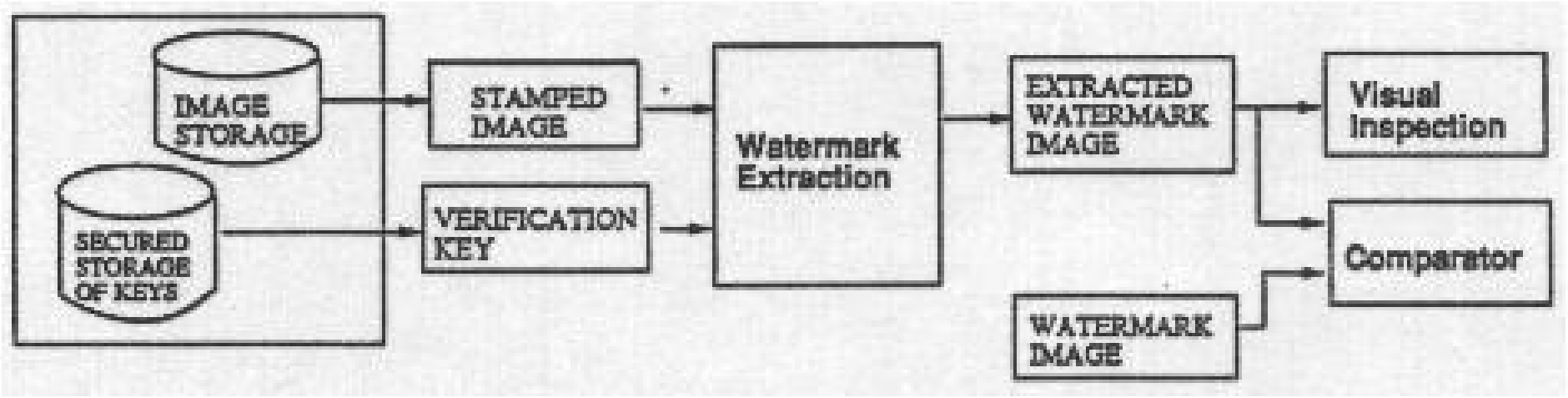
Wasserzeichen und robuste Hash Funktionen können auch kombiniert werden: Hash Werte können mit Wasserzeichen Technologie in die Daten eingebettet werden. In diesem Fall wird aber ein robustes Wasserzeichen Verfahren (wie für DRM) benötigt.

Fragiles Watermarking zur Integritätsprüfung (IBM) I

Das beschriebene Verfahren wurde von Yeung-Minzter [13] entwickelt. 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 (blind Verfahren) ! 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.

Fragiles Watermarking zur Integritätsprüfung (IBM) II



Fragiles Watermarking zur Integritätsprüfung (IBM) III

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:

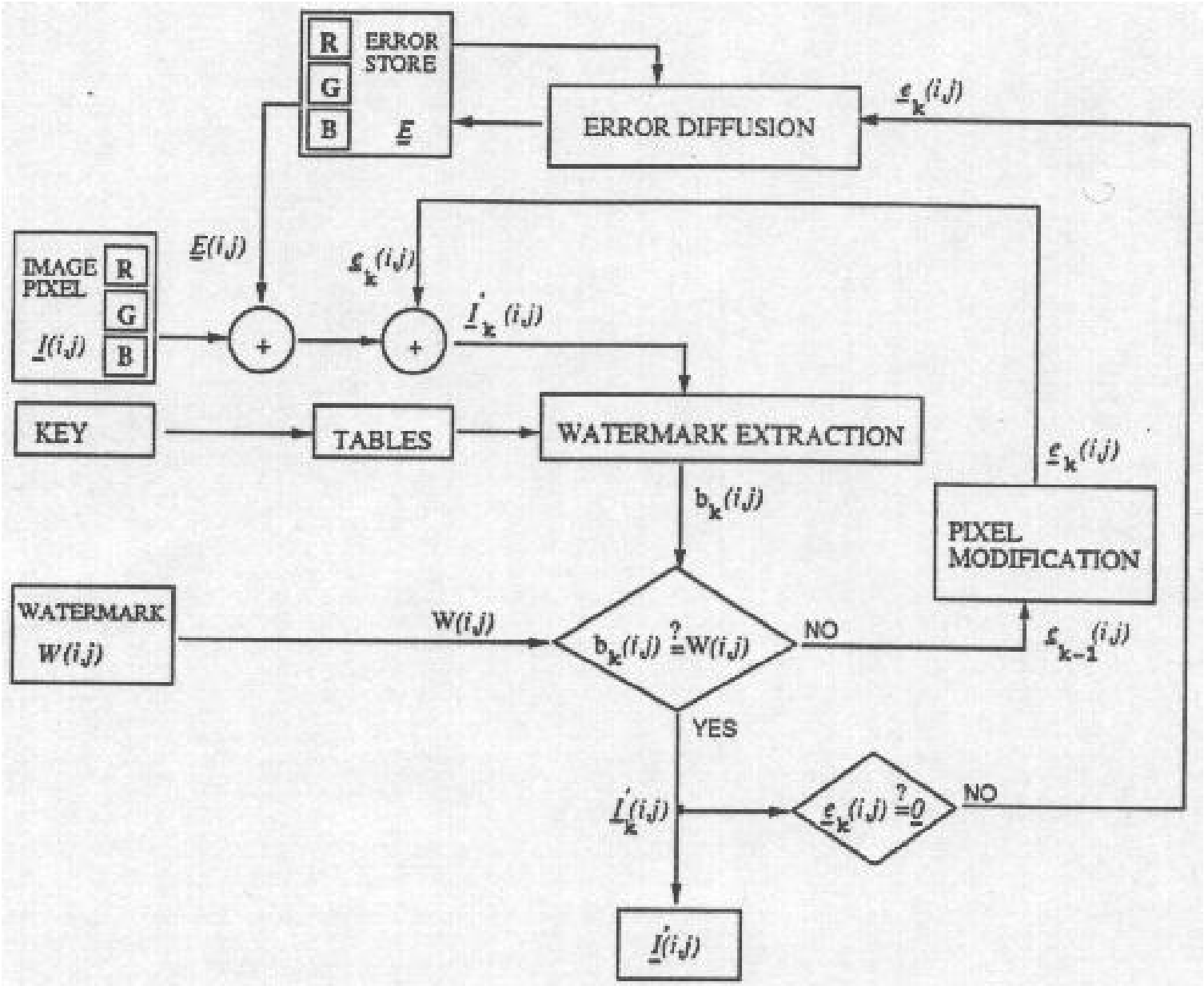
- Watermark Extraction: Die Aufgabe dieser Funktion ist es, des Farbwert des behandelten Bildpunktes in einen schwarz/weiss-Wert abzubilden. Die Funktion basiert auf einem Key (Tabellen), der z.B. zufaellig erzeugt werden kann. Die Ergebnisse jedes Farbkanals werden mit XOR verknüpft.

- 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.

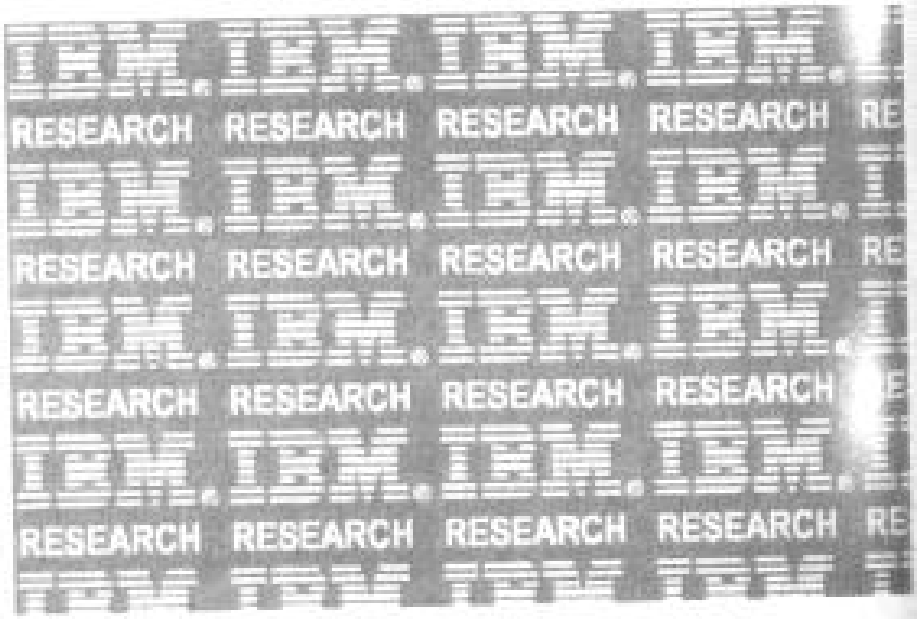
Fragiles Watermarking zur Integritätsprüfung (IBM) VI

- 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 durchgef”uhrten 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.

Fragiles Watermarking zur Integritätsprüfung (IBM) V



Fragiles Watermarking zur Integritätsprüfung (IBM) VI



Fragiles Watermarking zur Integritätsprüfung (IBM) VII

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

Fragiles Watermarking zur Integritätsprüfung (IBM) VIIIa

Angriffe gegen das Verfahren von IBM:

- Die effektivsten Attacken setzen voraus, dass derselbe Schlüssel und dasselbe Logo für mehrere Bilder verwendet wurden. In diesem Fall gilt für die Bilder u, v:

$$W(i,j) = f_g(u(i,j)) = f_g(v(i,j)) \forall (i,j) \tag{7}$$

  Aus dieser Gleichung lassen sich Gruppen von Grauwerten finden, die von f auf denselben Wert abgebildet werden. Innerhalb dieser Gruppen können Pixel ausgetauscht werden ohne das Wasserzeichen zu verändern. Bereits aus 2 Bildern lassen sich so ca. 90% von f rekonstruieren. Neben der Verwendung desselben Schlüssels und Logos fr mehrere Bilder setzt dieser Angriff die Unabhängigkeit von f und der Pixelposition (i,j) voraus. Benutzt man stattdessen eine Funktion $f(u(i,j),i,j)$ wird dieser Angriff unmöglich.

Fragiles Watermarking zur Integritätsprüfung (IBM) VIIIb

Weitere Angriffe gegen das Verfahren von IBM:

- Holliman-Memon (Collage) Attacke: geg. sei eine Menge von Bilder $I_1, I_2, \ldots, I_n$ die mit dem selben Schlüssel und Logo authentifiziert wurden. Aus diesen wird ein Bild J synthetisiert, sodass:

$$\forall i, jJ(i,j) = I_k(i,j)k \in 1, \ldots, n \tag{8}$$

  Die Qualität von J hängt von der Menge der verfügbaren Bilder ab. Als Gegenmassnahme wurde eine bildabhängige Funktion f vorgeschlagen (Probleme beim Detekten der Watermark im modifizierten Bild), alternativ kann f neben dem aktuellen Pixel auch deren Nachbarn miteinbeziehen (Verschlechterung der Lokalisation).

Fragiles Watermarking zur Integritätsprüfung (Alg. v. Wong) IX

1. Das Bild wird in 8x8 Pixel große, nicht überlappende Blöcke Z_i aufgeteilt. Jeder Block wird unabhängig weiterbearbeitet.

2. Ein binäres Logo wird periodisch wiederholt bis es die gane Bildfläche abdeckt und dann ebenfalls in 8x8 Blöcke A_i aufgeteilt.

3. Fr jeden Block Z_i ist Z_i^* der Block der durch Löschen der LSB entsteht. $H_i = Hash(Z_i^*)$

4. $H_i' = H_i \oplus A_i$

5. H_i' wird mit einem privaten Schlssel k verschlsselt, Ergebnis ist die Blocksignatur $S_i = E_k(H_i')$

6. S_i wird in die LSBs von Z_i^* eingefgt.

Fragiles Watermarking zur Integritätsprüfung (Alg. v. Wong) X

Angriffe gegen das Verfahren von Wong:

- Collage attack: hat ein Angreifer Zugang zu mehreren Bildern die mit dem selben Logo und dem selben Key authentifiziert wurden, so kann er aus den Blöcken der verschiedenen Bilder ein neues Bild zusammensetzen. Dies gilt insbesondere auch für Blöcke eines Bildes, die frei umherbewegt werden können. Diese Attacke wird durch die Unabhängigkeit von Blockposition und Blocksignatur ermöglicht und kann durch eine positionsabhängige Signatur erschwert werden. Weiters könnten neben dem Block Z_i auch benachbarte Blocks in den Hashwert H_i einbezogen werden, was allerdings zu einer Verschlechterung der Lokalisierung führt.

- Birthday Attacke: die Suche nach Blöcken mit gleichem Hashwert wird durch den auf 64 Bit beschränkten Hashwert ermöglicht. Die Beschränkung auf 64 Bit ist nötig weil der Hash in die LSB der Blöcke passen muss. Das Problem kann durch die Verwendung grösserer Blöcke gelöst werden, was allerdings wiederum zu einer Verschlechterung der Lokalisierung führt.

Semi-Fragiles Watermarking zur Integritätsprüfung I

Verfahren für semifragile Wasserzeichen lassen sich generell in 2 Schritte unterteilen. Die Erstellung der Wasserzeichen Information aus den Bilddaten oder einem Schlüssel und die Einbettung dieser Daten in das Bildmaterial.

Hier ergeben sich 2 unterschiedliche Vorgehensweisen:

- Lokale Authentifizierung: Die Erstellung des Wasserzeichens und die Einbettung erfolgt unabhängig für einzelne Blocks. Die Lokalisierung von Veränderungen erfolgt durch Authentifizierung jedes einzelnen Blocks.

- Globale Authentifizierung: Das Wasserzeichen beschreibt globale Bildeigenschaften und wird in das ganze Bild eingebettet. Veränderungen werden durch Vergleich des extrahierten Wasserzeichens mit dem vorhandenen Bild erkannt. Als Wasserzeichen wird häufig eine verkleinerte Variante des Bildes genutzt. Die Erstellung des Wasserzeichens zielt darauf ab, die wesentlichen Charakteristika des Bildes zu extrahieren und in einer geringen Anzahl von Bits darzustellen. Der Vorteil dieser Variante ist, das nach Extraktion des Wasserzeichens neben automatisierten Vergleichen auch ein optischer Vergleich mit dem Bild möglich ist. Darüber hinaus können aus den eingebetteten Daten veränderte Bereiche innerhalb des Bildes zumindest annähernd wiederhergestellt werden. Einbettung: robuste WM Verfahren.

Semi-Fragiles Watermarking zur Integritätsprüfung (lokale Authentifizierung) II

Das Verfahren von Lin, Podichuk und Delp [7] dient als Beispiel für semifragiles Watermarking mit lokaler Authentifizierung.

- Für jeden 8x8 Pixel Block wird ein Wasserzeichen erzeugt. Dazu wird eine schlüsselabhängige Folge von gaussverteilten Pseudozufallszahlen $C_1, \ldots C_{35}$ erzeugt. Diese werden als die niederfrequenten AC Koeffizienten eines DCT Blocks betrachtet. Der DC Koeffizient wird auf 0 gesetzt, um sichtbare Veränderungen des Bildes zu vermeiden. Die hochfrequenten AC Koeffizienten werden nicht genutzt, da sie bei einer JPEG Komprierimierung stark verändert werden.

- Die IDCT Transformation des DCT Blocks liefert das Wasserzeichen W.

- Aus W und dem Originalblock X entsteht der signierte Block Y. Der Parameter σ gibt die Stärke des Wasserzeichens an.

$$Y = X + \sigma W \tag{9}$$

Semi-Fragiles Watermarking zur Integritätsprüfung (lokale Authentifizierung) III

Angriff gegen das System:

- ist ein Angreifer im Besitz eines Originalbildes und eines gewatermarkten Bildes, kann er das Wasserzeichen einfach extrahieren und für das Signieren von anderen Bildern wiederverwenden. Das kann durch die Verwendung von Image features oder von bildahbhängigen Wasserzeichen verhindert werden (z.B. bei der Generierung der zufälligen Koeffizienten).

Die meisten Verfahren mit lokaler Authentifizierung sind zwar ausreichend stabil gegenüber Kompression und Rauschen und haben gute Lokalisationseigenschaften, geometrische Transformationen hingegen zerstören die Blockstruktur der Verfahren und sind so die “natürlichen Feinde”.

Robuste Multimedia Hash Funktionen I

Robuste Multimedia Hashes werden für verschiedene Anwendungsgebiete verwendet, unter anderem für Authentifizierung und die Suche in Bilddatenbanken. Diese beiden Anwendungsgebiete stellen unterschiedliche Anforderungen an die Verfahren. Zunächst allerdings die gemeinsamen Anforderungen:

- Gleichverteilung

$$Pr[H_K(X)] = \alpha] \approx \frac{1}{2^L}, \forall \alpha \in 0,1^L. \tag{10}$$

- Paarweise Unabhängigkeit bei visuell verschiedenen Eingangsbildern

$$Pr[H_K(X) = \alpha | H_K(Y) = \beta] \approx Pr[H_K(X)], \forall \alpha, \beta \in 0,1^L. \tag{11}$$

- Invarianz bei visueller Ähnlichkeit

$$Pr[H_K(X) = H_K(X')] \approx 1 \tag{12}$$

Robuste Multimedia Hash Funktionen IIa

Invarianz bei visueller Ähnlichkeit liefert die Robustheit. Die Ähnlichkeit hängt allerdings vom HVS und semantischen Eigenschaften des Bildes ab und ist nicht leicht zu quantifizieren. Um möglichst effektive Hashfunktionen zu konstruieren müssen diese auf Bildeigenschaften basieren, die sowohl invariant gegenüber häufigen Transformationen sind, als auch den fr den Menschen sichtbaren Bildinhalt gut repräsentieren.

Unterschiede durch das Anwendungsgebiet ergeben sich bei der Resistenz gegenüber intentionellen Änderungen.

- Authentifizierung: es soll möglichst schwierig sein, zu einem gegebenen Bild X ein visuell verschiedenes Bild X' zu finden, sodass $H(X) = H(X')$. Darunter fallen sichtbare Manipulationen am Bild X, wie etwa das Ausschneiden oder Hinzufügen von Objekten.

Robuste Multimedia Hash Funktionen IIb

- Bilddatenbanken: es soll die Konstruktion eines visuell ähnlichen Bildes X' mit $H(X) \neq H(X')$ möglichst aufwendig sein. Dies spielt beispielsweise bei der automatisierten Suche nach nicht lizensierten Bildkopien eine wesentliche Rolle.

Zusätzlich ist oft erwünscht, dass die Hammingdistanz zwischen Hashwerten als Mass für das Ausmass des Unterschiedes verwendet werden kann. Das steht wegen der Ermöglichung systematischer Attacken im Widerspruch zur Sicherheit !! Ein weiteres Problem für die Sicherheit vieler Hash Funktionen ist, dass die vom Algorithmus genutzten Bildeigenschaften öffentlich bekannt sind (z.B. Paare von DCT Koeffizienten, Mittelwerte, Mediane), was eine systematische Manipulation des Bildes vereinfacht.

Robuste Multimedia Hash Funktionen zur Authentifizierung I

Fridrich und Goljan [12] schlagen ein Verfahren vor, bei dem die Daten zur Hasherstellung nicht öffentlich bekannt sind. Genutzt wird, dass die niederen DC Koeffizienten eines Bildes gegenüber den meisten Operationen invariant sind und eine Veränderung derselben zu sichtbaren Artefakten im Bild führt.

- Aus einem geheimen Schlüssel K werden N zufällige Matrizen $M^{(i)}$ erzeugt, deren Einträge im Intervall $[0..1]$ gleichverteilt sind.
- Ein Lowpass Filter wird mehrmals auf jede Matrix $M^{(i)}$ angewandt. Das Ergebnis sind N glatte Patterns $P^(i)$.
- Von jedem Pattern $P^(i)$ wird dessen Mittelwert subtrahiert, womit der DC Koeffizient 0 gesetzt wird.
- Die Hashbits b_i ergeben sich durch Projektion von Bildblöcken auf die Patterns $P^{(i)}$:

$$if|B.P^{(i)}| < Thb_i = 0if|B.P^{(i)}| \geq Thb_i = 1 \tag{13}$$

Th ist dabei ein konstanter Threshold.

Robuste Multimedia Hash Funktionen zur Authentifizierung II

Venkatesan et al. (Microsoft Research) schlagen folgendes Verfahren vor:

- Das Bild wird Wavelet-transformiert.
- Für jedes Subband wird ein Featurevektor F_i berechnet. Dazu wird das Subband zufällig unterteilt und für jede Region ein statistisches Mass berechnet. Für das LL Band der Durchschnitt, ansonsten die Varianz.
- Die Gleitkommawerte von jedem F_i werden durch randomized rounding auf $0, \ldots, 7$ abgebildet. Das Ergebnis formt den vorläufigen Hashstring H_p.
- Auf H_p wird ein Reed-Muller error-correcting code im Dekoder Modus angewendet, was ihn kürzt und robuster macht.
- Im letzten Schritt wird noch ein linearer Code angewendet, der das Ergebnis nochmals kürzt und die Robustheit erhöht.

Robuste Multimedia Hash Funktionen zur Authentifizierung III

In Meixner und Uhl (SPIE WM & MMSec 2004) wird eine Attacke gegen das beschriebene System vorgestellt:

- Für ein gegebenes Bild I wird zuerst das manipulierte Bild F erstellt.
- I und F werden Wavelet transformiert.
- Für I und F wird eine identische Partition $I_1, \ldots, I_n$ und $F_1, \ldots, F_n$ erstellt.
- Je nachdem um welchen Typ Subband es sich handelt: $\forall$ Regionen i wird Durchschnitt oder Varianz in F_i so angepasst, dass der Wert in F_i dem Wert in I_i gleich oder ähnlich ist.
- F wird invers Wavelet transformiert um die gewünschte Fälschung zu erhalten.

Es stellt sich heraus, dass die zufällige Unterteilung im Algorithmus kaum Schutz gegen Angriffe bietet, weil durch eine ausreichende Grösse von n diese Unterteilung nicht bekannt sein muss. Das Ausmass der Modifikation beeinflußt natürlich die Qulaität aber auch die Ähnlichkeit der Hashwerte.

Robuste Multimedia Hash Funktionen zur Authentifizierung IV

Das gefälschte Bild hat zum Original eine Hammingdistanz von unter 0.02, gegenüber 0.1 bei JPEG quality 90.

Robuste Multimedia Hash Funktionen zur Datenbanksuche Ia

Mihcak und Venkatesan [40] schlagen für diese Applikation das folgende Verfahren vor:

- Auf das Ausgangsbild X wird eine DWT mit L Stufen angewandt. X_A ist das resultierende Approximation Subband (LL).

- X_A wird durch Thresholding zu einem binren Bild M:

$$M_1(i,j) = \begin{cases} 1 X_A(i,j) \geq T \\ 0 sonst \end{cases} \tag{14}$$

- Sei $S_{[m,n],p}(A)$ ein statistischer Ordnungsfilter, mit: $S_{[m,n],p}(A) = B$, wobei $\forall i, jB(i,j)$ das p-te Element der aufsteigend sortierten Menge $A(i',j') : i' \in i-m,\ldots,i+m, j' \in j-n,\ldots,j+n$ ist.

  f ist eine shiftinvariante Filterfunktion.

Robuste Multimedia Hash Funktionen zur Datenbanksuche Ib

- $M_2 = S_{[m,n],p}(M_1)$
- $M_3 = A.M_2$, $M_4 = f(M_3)$
- Aus M_4 wird durch Thresholding wieder ein Binrbild M_5 erzeugt.
- Ist die Rekursiontiefe $> C$ wird diese abgebrochen, Ansonsten wird die Differenz von M_5 und M_1 berechnet. Ist $D(M_1, M_5) < \varepsilon$ wird die Rekursion ebenfalls abgebrochen, ansonsten setzt sie mit Schritt 3 fort.
- $H = M_5$

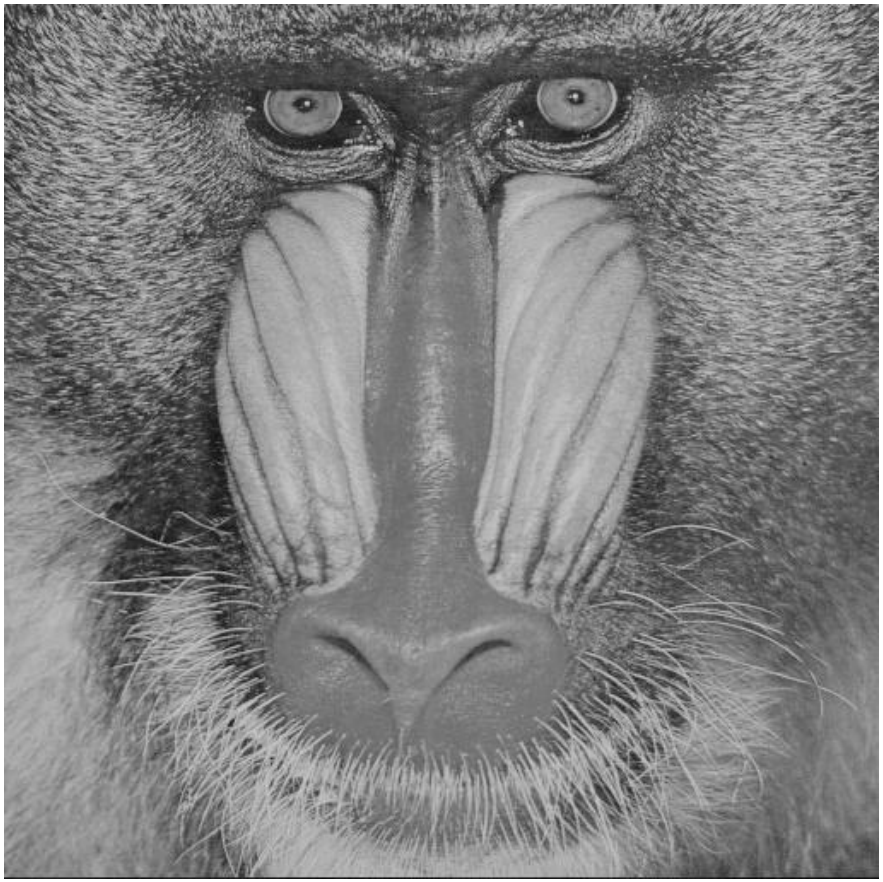
Um den Algorithmus schlüsselabhängig zu machen, wird das Bild zunächst zufällig in Rechtecke R_i eingeteilt und der Ausgangsalgorithmus auf alle Regionen angewandt. Die erzeugten Hashbits werden in zufälliger Reihenfolge konkateniert und danach ein Subset ausgewählt. Dieses Subset ist der finale Hashwert $H_K(X)$.

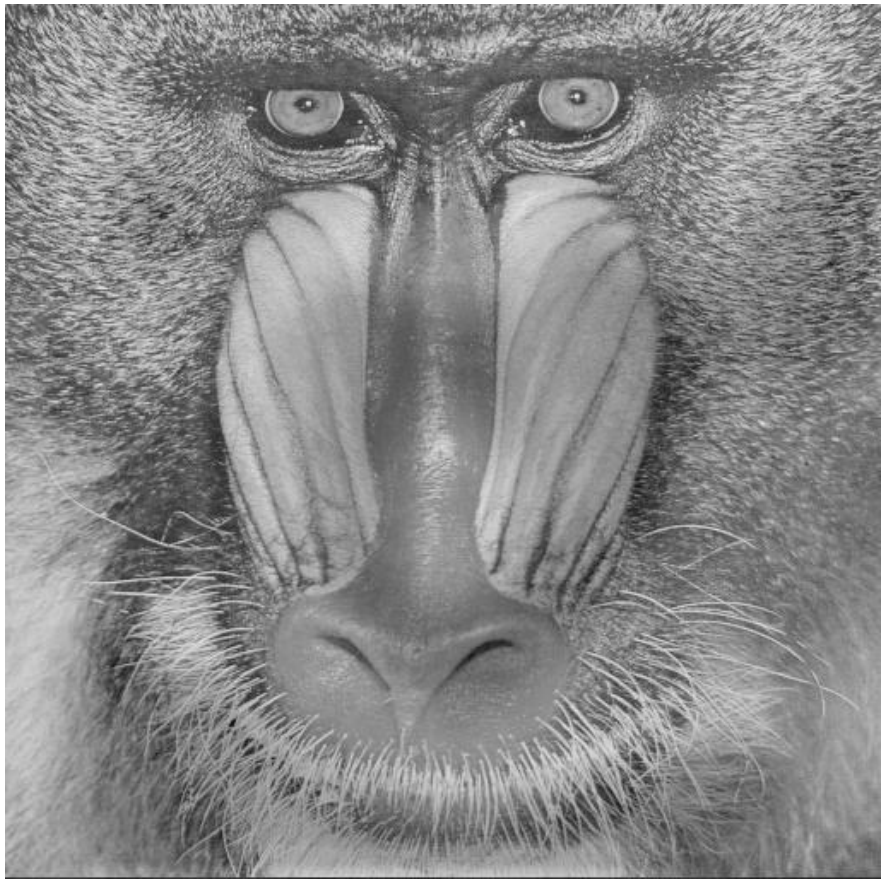
Robuste Multimedia Hash Funktionen zur Datenbanksuche II

Ein Angriff auf einen Datenbankhash versucht ein Bild so zu modifizieren, dass es einen möglichst unterschiedlichen Hash liefert, der visuelle Eindruck aber gleich bleibt. Das vorgestellte Verfahren ist gegenüber gewöhnlichen Modifikationen sehr stabil - der beste Angriffspunkt ist die Generierung des ersten S/W Bildes durch Thresholing. In diesem Schritt wird der Median als Threshold verwendet um eine gleiche Anzahl von S/W Pixeln zu erhalten.

Bei einem Angriff wird das Bild DWT transformiert und der Median des LL Subbands bestimmt. Im LL Subband werden dann Koeffizienten die nahe am Median liegen verändert (das verändert visuell kaum etwas, hat aber grossen Einfluss auf das Threshold Bild), dann wird das gefälschte Bild durch IDWT erzeugt.

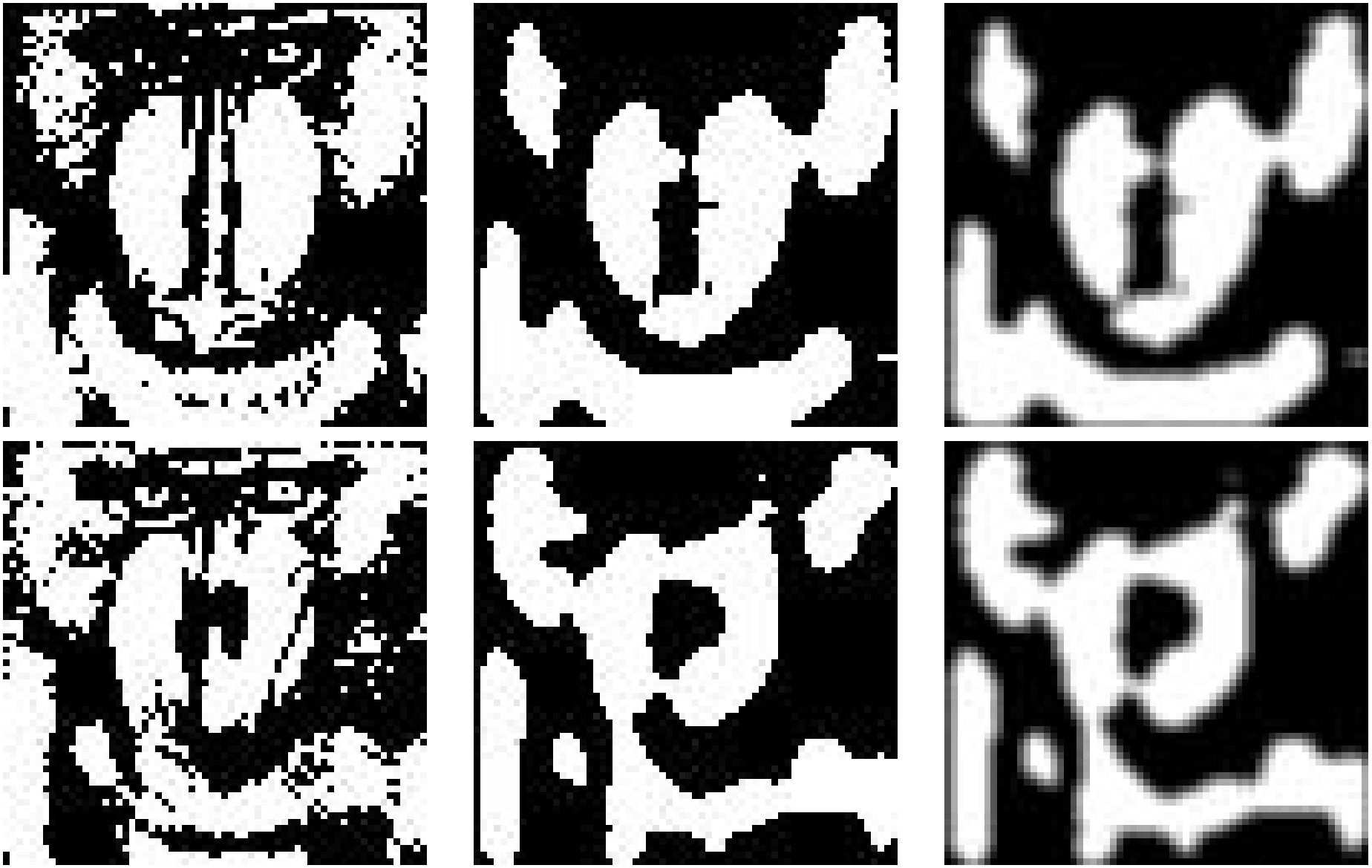
Robuste Multimedia Hash Funktionen zur Datenbanksuche III





Das gefälschte Bild hat zum Original eine Hammingdistanz von 0.35, Baboon hat zum Bild “Truck” eine Distanz 0.38 !!

Robuste Multimedia Hash Funktionen zur Datenbanksuche IV



Finale

Ich hoffe ich habe mit dieser LVA ihr Interesse an der Thematik wecken können. Ich wünsche ihnen schöne Ferien und alles Gute bei der Klausur.

Literaturhinweise

Literatur

[1] I. Agi and L. Gong. An empirical study of secure MPEG video transmissions. In *ISOC Symposium on Network and Distributed Systems Security*, pages 137–144, San Diego, California, 1996.

[2] Fadi Almasalha, Nikita Agarwal, and Ashfaq Khokhar. Secure multimedia transmission over RTP. In *Proceedings of the Eighth IEEE International Symposium on Multimedia (ISM'08)*, Berkeley, CA, USA, December 2008. IEEE Computer Society.

[3] H. Cheng and X. Li. On the application of image decomposition to image compression and encryption. In P. Horster, editor, *Communications and Multimedia Security II, IFIP TC6/TC11 Second Joint Working Conference on Communications and Multimedia Security, CMS '96*, pages 116–127, Essen, Germany, September 1996. Chapman & Hall.

[4] H. Cheng and X. Li. Partial encryption of compressed images and videos. *IEEE Transactions on Signal Processing*, 48(8):2439–2451, 2000.

[5] Ingemar J. Cox, Matthew L. Miller, and Jeffrey A. Bloom. *Digital Watermarking*. Morgan Kaufmann, 2002.

[6] Scott A. Craver, Nasir Memon, Boon-Lock Yeo, and Minerva M. Yeung. Can invisible watermarks resolve rightful ownerships? In Ishwar K. Sethi and Ramesh C. Jain, editors, *Proceedings of SPIE, Storage and Retrieval for Image and Video Databases*, volume 3022, pages 310–321, San Jose, CA, USA, July 1996.

[7] Edward J. Delp, Christine I. Podilchuk, and Eugene T. Lin. Detection of image alterations using semi-fragile watermarks. In Ping Wah Wong and Edward J. Delp, editors, *Proceedings of IS&T/SPIE's 12th Annual Symposium, Electronic Imaging 2000: Security and Watermarking of Multimedia Content II*, volume 3971, San Jose, CA, USA, January 2000.

[8] Werner Dietl, Peter Meerwald, and Andreas Uhl. Protection of wavelet-based watermarking systems using filter parametrization. *Signal Processing (Special Issue on Security of Data Hiding Technologies)*, 83(10):2095–2116, October 2003.

[9] Jana Dittmann, editor. *Digitale Wasserzeichen: Grundlagen, Verfahren, Anwendungsgebiete*. Springer Verlag, 2000.

[10] Jana Dittmann and Ralf Steinmetz. Enabling technology for the trading of MPEG-

encoded video. In *Information Security and Privacy: Second Australasian Conference, ACISP '97*, volume 1270, pages 314–324, July 1997.

[11] Dominik Engel, Rade Kutil, and Andreas Uhl. A symbolic transform attack on lightweight encryption based on wavelet filter parameterization. In *Proceedings of ACM Multimedia and Security Workshop, MM-SEC '06*, pages 202–207, Geneva, Switzerland, September 2006.

[12] Jiri Fridrich. Visual hash for oblivious watermarking. In Ping Wah Wong and Edward J. Delp, editors, *Proceedings of IS&T/SPIE's 12th Annual Symposium, Electronic Imaging 2000: Security and Watermarking of Multimedia Content II*, volume 3971, San Jose, CA, USA, January 2000.

[13] Jiri Fridrich, Miroslav Goljan, and Nasir Memon. Further attacks on yeung-mintzer fragile watermarking schemes. In Ping Wah Wong and Edward J. Delp, editors, *Proceedings of IS&T/SPIE's 12th Annual Symposium, Electronic Imaging 2000: Security and Watermarking of Multimedia Content II*, volume 3971, San Jose, CA, USA, January 2000.

[14] Raphaël Grosbois, Pierre Gerbelot, and Touradj Ebrahimi. Authentication and access control in the JPEG2000 compressed domain. In A.G. Tescher, editor, *Applications of Digital Image Processing XXIV*, volume 4472 of *Proceedings of SPIE*, pages 95–104, San Diego, CA, USA, July 2001.

[15] Michael Gschwandtner. Efficient protection and access control of 3d geometry data. Master's thesis, Department of Scientific Computing, University of Salzburg, Austria, March 2009.

[16] Michael Gschwandtner and Andreas Uhl. Toward DRM for 3D geometry data. In Edward J. Delp III, Ping Wah Wong, Jana Dittmann, and Nasir D. Memon, editors, *Proceedings of SPIE, Security, Forensics, Steganography, and Watermarking of Multimedia Contents X*, volume 6819, page 68190V ff., San Jose, CA, USA, January 2008. SPIE.

[17] Neil F. Johnson, Zoran Duric, and Sushil Jajodia. *Information Hiding: Steganography and Watermarking - Attacks and Countermeasures*. Kluwer Academic Publishers, 2000.

[18] Stefan Katzenbeisser and Fabien A. P. Petitcolas. *Information Hiding Techniques for Steganography and Digital Watermarking*. Artech House, December 1999.

[19] Thomas Kunkelmann. Applying encryption to video communication. In *Proceedings of the Multimedia and Security Workshop at ACM Multimedia '98*, pages 41–47, Bristol, England, September 1998.

[20] R. Norcen, M. Podesser, A. Pommer, H.-P. Schmidt, and A. Uhl. Confidential sto-

rage and transmission of medical image data. *Computers in Biology and Medicine*, 33(3):277 – 292, 2003.

[21] M. Podesser, H.-P. Schmidt, and A. Uhl. Selective bitplane encryption for secure transmission of image data in mobile environments. In *CD-ROM Proceedings of the 5th IEEE Nordic Signal Processing Symposium (NORSIG 2002)*, Tromso-Trondheim, Norway, October 2002. IEEE Norway Section. file cr1037.pdf.

[22] A. Pommer and A. Uhl. Wavelet packet methods for multimedia compression and encryption. In *Proceedings of the 2001 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, pages 1–4, Victoria, Canada, August 2001. IEEE Signal Processing Society.

[23] A. Pommer and A. Uhl. Application scenarios for selective encryption of visual data. In J. Dittmann, J. Fridrich, and P. Wohlmacher, editors, *Multimedia and Security Workshop, ACM Multimedia*, pages 71–74, Juan-les-Pins, France, December 2002.

[24] A. Pommer and A. Uhl. Selective encryption of wavelet packet subband structures for obscured transmission of visual data. In *Proceedings of the 3rd IEEE Benelux Signal Processing Symposium (SPS 2002)*, pages 25–28, Leuven, Belgium, March 2002. IEEE Benelux Signal Processing Chapter.

[25] A. Pommer and A. Uhl. Selective encryption of wavelet packet subband structures for secure transmission of visual data. In J. Dittmann, J. Fridrich, and P. Wohlmacher, editors, *Multimedia and Security Workshop, ACM Multimedia*, pages 67–70, Juan-les-Pins, France, December 2002.

[26] A. Pommer and A. Uhl. Selective encryption of wavelet-packet encoded image data — efficiency and security. *ACM Multimedia Systems (Special issue on Multimedia Security)*, 9(3):279–287, 2003.

[27] Lintian Qiao and Klara Nahrstedt. Comparison of MPEG encryption algorithms. *International Journal on Computers and Graphics (Special Issue on Data Security in Image Communication and Networks)*, 22(3):437–444, 1998.

[28] Josef Scharinger. Robust watermark generation for multimedia copyright protection. In *Proceedings of IWSSIP '99*, pages 177–180, Bratislava, Slovakia, 1999.

[29] J. Schneid and S. Pittner. On the parametrization of the coefficients of dilation equations for compactly supported wavelets. *Computing*, 51:165–173, May 1993.

[30] C. Shi and B. Bhargava. A fast MPEG video encryption algorithm. In *Proceedings of the Sixth ACM International Multimedia Conference*, pages 81–88, Bristol, UK, September 1998.

[31] S.U. Shin, K.S. Sim, and K.H. Rhee. A secrecy scheme for MPEG video data using the joint of compression and encryption. In *Proceedings of the 1999 Information Security Workshop (ISW'99)*, volume 1729 of *Lecture Notes on Computer Science*, pages 191–201, Kuala Lumpur, November 1999. Springer-Verlag.

[32] Champskud J. Skrepth and Andreas Uhl. Selective encryption of visual data: Classification of application scenarios and comparison of techniques for lossless environments. In B. Jerman-Blazic and T. Klobucar, editors, *Advanced Communications and Multimedia Security, IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security, CMS '02*, pages 213 – 226, Portoroz, Slovenia, September 2002. Kluver Academic Publishing.

[33] G. Spanos and T. Maples. Performance study of a selective encryption scheme for the security of networked real-time video. In *Proceedings of the 4th International Conference on Computer Communications and Networks (ICCCN'95)*, Las Vegas, NV, 1995.

[34] Thomas Stütz and Andreas Uhl. On format-compliant iterative encryption of JPEG2000. In *Proceedings of the Eighth IEEE International Symposium on Multimedia (ISM'06)*, pages 985–990, San Diego, CA, USA, December 2006. IEEE Computer Society.

[35] Thomas Stütz and Andreas Uhl. (In)secure multimedia transmission over RTP. In

Proceedings of the 18th European Signal Processing Conference, EUSIPCO '10, Aarlborg, Danmark, August 2010. EURASIP.

[36] L. Tang. Methods for encrypting and decrypting MPEG video data efficiently. In *Proceedings of the ACM Multimedia 1996*, pages 219–229, Boston, USA, November 1996.

[37] Ali Saman Tosun and Wu chi Feng. On error preserving encryption algorithms for wireless video transmission. In *Proceedings of the ninth ACM Multimedia Conference 2001*, pages 302–307, Ottawa, Canada, October 2001.

[38] T. Uehara and R. Safavi-Naini. Chosen DCT coefficients attack on MPEG encryption schemes. In *Proceedings of the 2000 IEEE Pacific Rim Conference on Multimedia*, pages 316–319, Sydney, December 2000. IEEE Signal Processing Society.

[39] T. Uehara, R. Safavi-Naini, and P. Ogunbona. Securing wavelet compression with random permutations. In *Proceedings of the 2000 IEEE Pacific Rim Conference on Multimedia*, pages 332–335, Sydney, December 2000. IEEE Signal Processing Society.

[40] Ramarathnam Venkatesan and M. Kivanc Mihcak. New iterative geometric methods for robust perceptual image hashing. In *Proceedings of the Workshop on*

Security and Privacy in Digital Rights Management 2001, Philadelphia, PA, USA, November 2001.

[41] Jiangtao Wen, Mike Severa, Wenjun Zeng, Max Luttrell, and Weiyin Jin. A format-compliant configurable encryption framework for access control of multimedia. In *Proceedings of the IEEE Workshop on Multimedia Signal Processing, MMSP '01*, pages 435–440, Cannes, France, October 2001.

[42] Jiangtao Wen, Mike Severa, Wenjun Zeng, Max Luttrell, and Weiyin Jin. A format-compliant configurable encryption framework for access control of video. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(6):545–557, June 2002.

[43] Tsung-Li Wu and S. Felix Wu. Selective encryption and watermarking of MPEG video (extended abstract). In Hamid R. Arabnia, editor, *Proceedings of the International Conference on Image Science, Systems, and Technology, CISST '97*, Las Vegas, USA, February 1997.

[44] Wenjun Zeng and Shawmin Lei. Efficient frequency domain video scrambling for content access control. In *Proceedings of the seventh ACM International Multimedia Conference 1999*, pages 285–293, Orlando, FL, USA, November 1999.