Digitale Wasserzeichenverfahren zur Überprüfung der Echtheit von Bildern

Dissertation

zur
Erlangung des akademischen Grades

Doktor-Ingenieur (Dr.-Ing.)

der Fakultät für Informatik und Elektrotechnik
der Universität Rostock

vorgelegt von
Mathias Schlauweg, geb. am 23. August 1979 in Rostock

Rostock, 05. Juni 2009

urn:nbn:de:gbv:28-diss2009-0202-6

Als Dissertation genehmigt von der

Fakultät für Informatik und Elektrotechnik

der Universität Rostock

Gutachter:

- Prof. Dr.-Ing. habil. Erika Müller, Universität Rostock
- Prof. Dr.-Ing. Jana Dittmann, O. v. Guericke-Universität Magdeburg
- Prof. Dr. Andreas Uhl, Universität Salzburg

Tag der Einreichung:

05. Juni 2009

Tag der öffentlichen Verteidigung:

13. November 2009

Vorwort

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Nachrichtentechnik der Fakultät für Informatik und Elektrotechnik der Universität Rostock. Sie widerspiegelt einen großen Teil meiner Forschungsergebnisse, die in den durch das Bundesland Mecklenburg-Vorpommern geförderten LfS-IuK-Projekten "Multimediales Content Management in Mobilen Umgebungen", "Proaktive Verteilte Informationssysteme" sowie "Mobile Assistenzsysteme" erzielt wurden. Darüber hinaus wurde die Arbeit auch durch meine Beschäftigung als Kollegiat des DFG-Graduiertenkollegs "Verarbeitung, Verwaltung, Darstellung und Transfer multimedialer Daten - technische Grundlagen und gesellschaftliche Implikationen" (GRK 466) unterstützt und positiv beeinflusst.

Für die fachliche Betreuung sowie die wertvollen Anregungen und Hinweise möchte ich mich bei meiner Mentorin Frau Prof. Dr.-Ing. habil. E. Müller recht herzlich bedanken.

Allen Mitarbeitern der Arbeitsgruppe Signal- und Bildverarbeitung danke ich für das angenehme und freundliche Arbeitsklima. Durch den fachlichen Gedankenaustausch wurde ein Teil zum Gelingen der Arbeit beigetragen.

Mein besonderer Dank gilt meiner Familie und meinen Freunden für die moralische Unterstützung und dauerhafte Ermutigung.

Kurzfassung

Im Laufe der letzten Jahre haben sich durch den Einsatz der Digitaltechnik im Bereich Multimedia zahlreiche Vorteile ergeben. Digitale Bilder können einfach und kostengünstig erstellt, ohne Qualitätsverluste vervielfältigt und ohne spezielle Kenntnisse verändert werden. Diese Eigenschaften können jedoch auch zu Nachteilen führen. Urheber- und Eigentümerrechte lassen sich nur schwer durchsetzen. Veränderungen der Daten sind oft nicht nachweisbar, und die Echtheit ist daher grundsätzlich anzuzweifeln.

Um die Echtheit zu überprüfen, ohne dabei die Gewohnheiten des Anwenders einzuschränken, können zusätzliche Daten als Digitale Wasserzeichen in die Bilder eingebettet werden.

Die zusätzlichen Daten sollen robust gegenüber erlaubten Nachbearbeitungen oder Konvertierungen eines Bildes in andere Kompressionsformate sein. Sie sollen jedoch bei der Überprüfung einen Alarm auslösen, wenn der Bildinhalt unerlaubt verändert wird. Zudem sollen die durch das Einbetten der Daten verursachten Verzerrungen des Bildes nicht wahrnehmbar und das Gesamtsystem nicht manipulierbar sein.

Diese Zielstellung wird durch bekannte Verfahren bisher nur unzureichend erfüllt. Die vorliegende Arbeit liefert einen Beitrag zur Entwicklung von Wasserzeichensystemen zur effizienten und manipulationssicheren Überprüfung der Echtheit von Bildern. Durch die Quantisierung der Koeffizienten der Diskreten Wavelet-Transformation wird, in den Prozess einer JPEG2000-Bildkompression integriert, manipulationssicher und nicht-wahrnehmbar ein an den Bildinhalt angepasstes Wasserzeichen eingebettet. Es ist robust gegenüber einer breiten Auswahl unterschiedlicher erlaubter Bildoperationen. Zu diesen erlaubten Operationen zählen die Kompression des Bildes, Helligkeits- und Kontraständerungen, Filterungen, Bildschärfungen sowie die Skalierung der Bildgröße.

Um ein Wasserzeichen darüber hinaus auch nach geometrischen Veränderungen des Bildes extrahieren zu können, wird in dieser Arbeit ein zweites Verfahren neu entwickelt. Die mit diesem Verfahren eingebetteten Wasserzeichendaten sind auch robust gegenüber Bildrotation, -verschiebung, -scherung und dem Entfernen weniger Spalten bzw. Zeilen am Bildrand.

Die Leistungsfähigkeit beider Wasserzeichenverfahren und der in diesem Zusammenhang entwickelten Resynchronisation von Wasserzeichendaten wird ausführlich analysiert und mit der Leistungsfähigkeit ähnlicher Verfahren anderer Autoren verglichen.

Abstract

During the last decade, growing applications of digital technologies in the field of multimedia resulted in various advantages. Digital images can be created easily and at a reasonable price. They can be copied without quality loss and changed without special knowledge. But, these properties can also yield disadvantages. For example, it is hard to assert rights of authors and owners and to proof the authenticity of images.

To verify the authenticity without limiting user's customs additional data can be embedded within images by means of digital watermarks.

Additionally embedded data should be robust against allowed image processing or compression format conversions. But, if the content of an image is tampered with, then an alarm should be raised during verification. Further, image distortions caused by data embedding should be imperceptible and it should be impossible to manipulate the overall system.

These objectives are not met by any known system, so far. For that reason, the present work contributes the development of digital watermarking systems for efficient and tamper-proof image authentication. A digital watermark adapted to the image content is embedded imperceptibly by quantization of the coefficients of the discrete wavelet transform domain. This process is directly integrated into a JPEG2000 image compression and hence very efficient. The embedded watermark is robust against a variety of allowed image processing operations, e.g., image compression, change of luminance and contrast, filtering, sharpening as well as scaling of image size.

To further enable watermark extraction after changes of image geometry a second method is developed, in this work. Using this method, embedded watermark data is also robust against image rotation, translation, skewing and cropping of few columns or rows at image margin.

The performances of both watermarking methods as well as a resynchronization of watermark data proposed in this context are extensively analyzed and compared to data of similar methods by other authors.

Inhaltsverzeichnis

Abkürzungen und Symbole

Abkürzungen

AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
BSC	Binary Symmetric Channel
bpp	Bit pro Pixel
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DWT	Discrete Wavelet Transform
EBCOT	Embedded Block Coding with Optimal Truncation
FEC	Forward Error Correction
FNR	False Negative Ratio
FPR	False Positive Ratio
HVS	Human Visual System
IDWT	Inverse Discrete Wavelet Transform
JPEG	Joint Photographic Experts Group
LABR	Lowest Authentication Bit Rate
LBM	Low Bit Modulation
LOG	Laplacian of the Gaussian
LPM	Log-Polar Mapping
LSB	Least Significant Bit
MER	Mask Error Rate
MSB	Most Significant Bit
PSNR	Peak Signal-to-Noise Ratio
QIM	Quantization Index Modulation
RA	Repeat-Accumulate
RS	Reed-Solomon

RCT	Reversible Color Transform
RGB	Farbkomponenten rot, grün, blau
ROI	Region of Interest
RST	Rotation, Scaling, Translation
SCS	Scalar Costa Scheme
TCQ	Trellis Coded Quantization
WZ	Wasserzeichen
YC_bC_r	Luminanz und Crominanzkomponenten

Mathematische Größen

Δ	Quantisierungsschrittweite
Λ	Quantisierungsgitter
A	Ergebnisbild der *LOG*-Filterung
B	Gray-Level-Blob
C	Sicherheit einer Entscheidung (engl. *certainty*)
d_m	Zitter-Vektor (engl. *dither*)
D	Kandidaten für Einfügungen D_I bzw. Auslöschungen D_E
F	Texturmerkmal zur Bildsegmentierung (engl. *feature*)
g	Wichtungsfaktor der Helligkeits- und Kontrastnormierung
I	Trägerbild im Grauwerte-Pixelbereich
$\mathfrak{H}$	Einweg-Hash-Funktion
h	Zusammensetzung mehrerer aneinandergereihter Hash-Werte
K	Anzahl der Einbettungspositionen, Länge des Wasserzeichens
L	Länge der einzubettenden Daten (ohne Fehlerkorrektur)
m	einzubettende Nachricht (ohne Fehlerkorrektur)
$\mathcal{M}$	Symbolalphabet
M	Metrik des Viterbi-Decoders
N	Länge des Trägersignals

O	Blob-Maskenmultiplikator
p	Einbettungs-/Extraktionsposition
q	quantisiertes Signal
Q	Quantisierer
S	Skalierungsfaktor einer Änderung der Bildgröße
σ	Standardabweichung bzw. Skale der *LOG*-Filterung
τ	Schwellwert
w	Digitales Wasserzeichen
x	Originalträgersignal
y	mit Wasserzeichen versehenes Trägersignal
γ	Einbettungsstärke
X	Bildbreite
Y	Bildhöhe
Z	Zerlegungsstufe der Wavelet-Transformation

Mathematische Symbole und Operatoren

$\bar{a}$	Mittelwert der Zahl a
&	binäre (bitweise) UND-Verknüpfung
|	binäre (bitweise) ODER-Verknüpfung
$\arg\min_a$	liefert diejenigen Parameterwerte a, die den Term minimieren
$\operatorname{sign}(a)$	Vorzeichen der Zahl a
$\lvert a \rvert$	Betrag der Zahl a bzw. Mächtigkeit der Menge a.
$\lVert a \rVert$	Norm des Vektors a. Entspricht Länge des Vektors, ähnlich der Betragsfunktion.
$\lfloor a \rfloor$	größter ganzzahliger Wert kleiner oder gleich a
$\lceil a \rceil$	kleinster ganzzahliger Wert größer oder gleich a
$[a,b]$	Wertepaar
$[a;b]$	geschlossenes Intervall von a bis b
$[a;b)$	nur linksseitig geschlossenes Intervall von a bis b

$(a;b)$ offenes Intervall von a bis b

Kapitel 1

Einleitung

1.1 Thematischer Hintergrund

Wenn man einen Euro-Geldschein gegen das Licht hält, wird in der lokal veränderten Papierstruktur eingebettet ein Wasserzeichenbild sichtbar. Es zeigt das auf dem Geldschein abgebildete, typische Motiv in einer zweiten, verkleinerten Darstellung und ist nur sehr schwer zu fälschen. Als ein Teil des Sicherheitssystems heutiger Bargeldzahlung dienen Wasserzeichen dazu, eine unerlaubte Vervielfältigung der Geldscheine zu verhindern.

In Analogie zum Wasserzeichen in Geldscheinen wurden in den frühen 90er-Jahren Digitale Wasserzeichen eingeführt, um Multimediadaten vor unerlaubter Vervielfältigung und Missbrauch zu schützen.

Digitale Multimediadaten (Bilder, Video, Audio) können einfach und kostengünstig erstellt, ohne Qualitätsverluste vervielfältigt und ohne spezielle Kenntnisse verändert werden. Einerseits vorteilig, können diese Eigenschaften auch zu Nachteilen führen. Urheber- und Eigentümerrechte lassen sich nur schwer durchsetzen. Veränderungen der Daten sind oft nicht nachweisbar, und die Echtheit ist daher grundsätzlich anzuzweifeln.

Das Einfügen eines Digitalen Wasserzeichens zum Schutz der Daten hat daher stetig steigendes Interesse erfahren.

Wie beim Einbetten eines Wasserzeichens im Geldschein kann auch die lokale Struktur eines Multimediasignals verändert werden. Beispielsweise können die Pixelfarbwerte eines Bildes oder Videos gezielt und nicht-wahrnehmbar verändert und auf diese Weise zusätzliche Daten als Wasserzeichen eingebettet werden. Ist der Prozess dieser gezielten Veränderung bekannt, lässt sich das versteckte Wasserzeichen fortan im markierten Multimediasignal nachweisen.

Die als Wasserzeichen eingebetteten Daten können Hinweise auf den Eigentümer, den Vertriebsweg (bzw. Käufer) der Multimediadaten oder auch den Inhalt zur Überprüfung der Echtheit der Daten sein.

Es wurden Systeme entwickelt, um Wasserzeichen nicht-wahrnehmbar und robust gegenüber Veränderungen des Multimediasignals als Träger der zusätzlichen Daten einzubetten. Kanalmodelle wurden erstellt, um eine theoretische Obergrenze für die Kapazität der einzubettenden Wasserzeichendaten aufzustellen. Angreifer haben immer wieder neue Manipulationsversuche unternommen und Systemdesigner haben immer wieder neue Gegenmaßnahmen erarbeitet. Die Entwicklung sicherer Digitaler Wasserzeichensysteme ist seitdem ein iterativer Prozess, ähnlich der viele Jahrzehnte währenden Entwicklung sicherer kryptografischer Verfahren.

1.2 Motivation und Zielstellung

Digitale Wasserzeichen müssen nicht-wahrnehmbar im Träger eingebettet sein. Der gewohnte Umgang mit den Multimediadaten soll nicht gestört werden. Ein Bild soll bspw. von einem in ein anderes Speicherformat konvertiert werden können, ohne dass spezielle Software eingesetzt werden muss. Das eingebettete Wasserzeichen soll diese Formatkonvertierung überstehen. Es soll auch dann nachweisbar sein, wenn das Bild nach der Aufnahme, wie gewohnt, skaliert, rotiert, in Hinblick auf Helligkeit, Kontrast oder Bildschärfe nachbearbeitet wurde.

Die Zielstellung, dass Digitale Wasserzeichen robust gegenüber einer breiten Auswahl von Nachbearbeitungen sein sollen, wird durch bekannte Verfahren bisher nur unzureichend erfüllt. Ein Wasserzeichen, mit dessen Hilfe die Echtheit eines Bildes überprüft werden soll, muss gegenüber allen Störungen robust sein, die den Bildinhalt nicht verändern. Werden jedoch Objekte im Bild durch Hintergrund ersetzt, aus unterschiedlichen Bildern vereint oder manipuliert und dadurch der Bildinhalt verändert, dann soll mithilfe des eingebetteten Wasserzeichens ein verlässlicher Alarm ausgelöst werden.

Ziel dieser Arbeit ist es, ein Digitales Wasserzeichenverfahren zur Überprüfung der Echtheit von Bildern zu entwickeln, welches die folgenden Kriterien erfüllt:

- die Einbettung der Wasserzeichendaten soll keine wahrnehmbaren Verzerrungen des Trägerbildes zur Folge haben,
- die Wasserzeicheneinbettung soll effizient sein, damit der Schutz der Echtheit bereits während der Bildaufnahme in der Kamera stattfinden kann,
- die durch eingebettete Wasserzeichen geschützten Bilder sollen, wie gewohnt, nachbearbeitet und verlustbehaftet in andere Kompressionsformate konvertiert werden können,
- das Wasserzeichensystem soll manipulationssicher sein, so dass jegliche Veränderung des Bildinhaltes eines geschützten Bildes erkannt werden kann.

1.3 Gliederung der Arbeit

Kapitel 2 gibt zunächst eine Einführung in die Grundlagen Digitaler Wasserzeichen. Dabei liegt der Fokus hauptsächlich auf der Erläuterung der in dieser Arbeit eingesetzten Techniken zur Einbettung von Wasserzeichendaten in Bildern. Da die Einbettung des ersten der beiden neu entwickelten Wasserzeichenverfahren direkt in den Prozess einer JPEG2000-Einzelbildkompression integriert ist, enthält das Grundlagenkapitel zudem eine Beschreibung wichtiger Teilsysteme dieses Kompressionsstandards. Darüber hinaus befasst sich Kapitel 2 mit der im Rahmen dieser Arbeit eingesetzten Faltungscodierung zur Fehlerkorrektur sowie dem um spezielle Algorithmen erweiterten Viterbi-Decoder.

In **Kapitel 3** wird ein System zur Überprüfung der Echtheit von Bildern entwickelt, dessen Komponenten direkt in den Prozess einer JPEG2000-Einzelbildkompression integriert sind. Dabei wird die aus dem Bildinhalt generierte Authentifizierungssignatur als Wasserzeichen in den Koeffizienten der Diskreten Wavelet-Transformation eingebettet. Zur Erweiterung der Leistungsfähigkeit werden Normierungen und Anpassungen der verwendeten Teilkomponenten beschrieben und untersucht. Das Kapitel wird mit einer Analyse der Gesamtleistungsfähigkeit und einem Vergleich mit Verfahren anderer Autoren abgeschlossen.

Anhand der Erkenntnisse des entworfenen Authentifizierungssystems erfolgt in **Kapitel 4** eine Erweiterung der JPEG2000-basierten Wasserzeicheneinbettung mit dem Ziel der Leistungsoptimierung. Die Einbettungsstärke wird an die Eigenschaften der lokalen Bildtextur angepasst. Im Zuge dessen werden in diesem Kapitel Verfahren zur Segmentierung eines Bildes in Bereiche unterschiedlicher Textureigenschaften entwickelt. Es kommt zu einer Erweiterung der eingesetzten Fehlerkorrektur. Die Leistungsfähigkeit des Gesamtsystems wird ausführlich analysiert und mit dem Verfahren aus Kapitel 3 verglichen.

Kapitel 5 dient der Beschreibung eines zweiten, in dieser Arbeit entwickelten, Wasserzeichenverfahrens auf der Grundlage so genannter Gray-Level-Blobs. Anhand der Erfahrungen durch den Entwurf und die Umsetzung des ersten, JPEG2000-basierten Verfahrens wird eine neue Domain zur Einbettung von Wasserzeichendaten entworfen. Ziel ist es, ein Verfahren zu entwickeln, dessen eingebettete Daten auch gegenüber geometrischen Veränderungen des Trägerbildes (bspw. Rotation, Verschiebung, Scherung) robust sind. Die im Kapitel 4 vorgestellten Normierungen und Anpassungen werden in diesem Kapitel ebenfalls eingesetzt.

Im Zuge der Implementierung und Evaluierung des in Kapitel 5 beschriebenen Wasserzeichenverfahrens stellte sich heraus, dass es bei diesem neuen Ansatz zu Desynchronisationsproblemen während der Wasserzeichenextraktion kommen kann. Aus diesem Grund wird die eingesetzte Fehlerkorrektur abermals erweitert, wobei die Algorithmen des Viterbi-Decoders zu Zwecken der Resynchronisation gezielt verändert werden. Die Beschreibungen dieser Ver-

änderung eines Viterbi-Decoders sowie der Erweiterungen des in Kapitel 5 entwickelten Wasserzeichenverfahrens befinden sich im **Kapitel 6**. Abschließend wird eine Leistungsanalyse des Gesamtsystems durchgeführt und die erreichte Leistungsfähigkeit mit den Daten von Verfahren anderer Autoren verglichen.

Der inhaltliche Aufbau dieser Arbeit wird durch Abbildung 1.1 veranschaulicht.

Kapitel 7 enthält eine Zusammenfassung der Arbeit und einen Ausblick auf zukünftige Aufgaben.

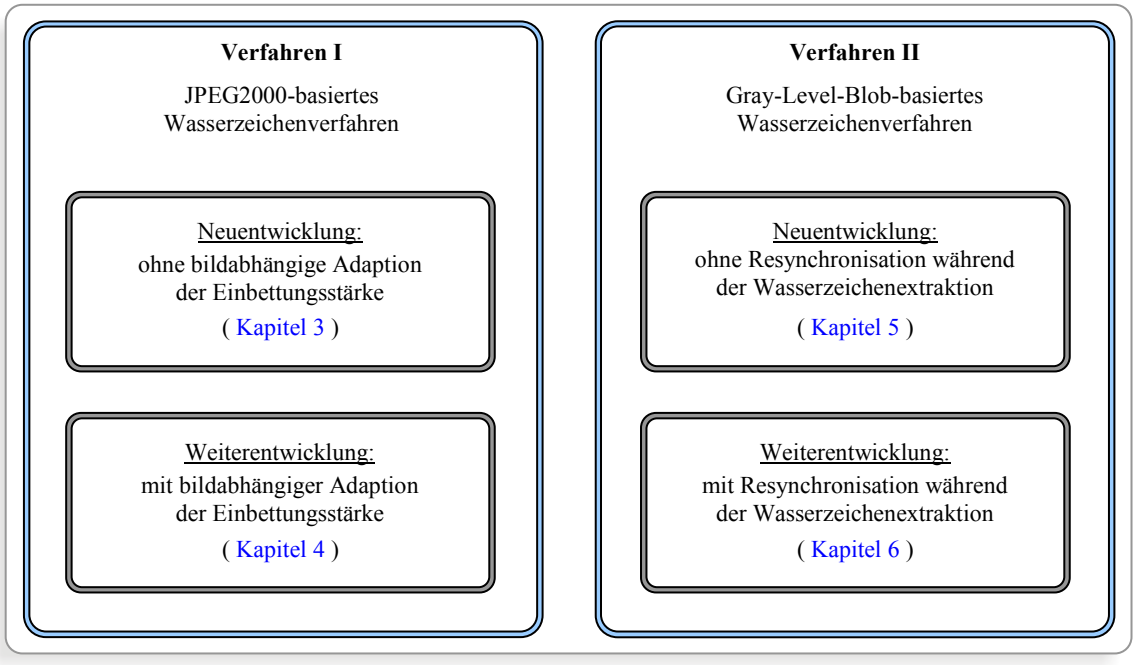


Abbildung 1.1 Übersicht der in dieser Arbeit entwickelten Wasserzeichenverfahren

Kapitel 2

Grundlagen

Gegenstand dieses Kapitels ist die Darstellung wesentlicher Grundlagen einiger in dieser Arbeit behandelter Thematiken. Zuerst erfolgt eine kurze Beschreibung von Aufbau, Funktion, Vor- und Nachteilen sowie Anwendungsgebieten Digitaler Wasserzeichensysteme. Daran anschließend gibt dieses Kapitel, als Basis für das in dieser Arbeit entwickelte Wasserzeichenverfahren, einen Einblick in die Bilddatenkompression JPEG2000. Zum Schluss findet eine Erläuterung des Viterbi-Decoders statt, der im Rahmen dieser Arbeit zur Fehlerkorrektur eingesetzt und zur Erlangung erweiterter Funktionalität gezielt verändert wird.

2.1 Digitale Wasserzeichen

Digitale Wasserzeichen sind in digitalen Medien versteckte, zusätzliche Daten. Diese digitalen Medien, zu denen Bilder, Videos, Audiodaten und auch Text zählen, dienen den zusätzlichen Daten als Träger. Das Wasserzeichen wird nicht in den Metadaten, also einem speziellen Datenblock einer Datei, untergebracht. Es ist direkt mit dem Inhalt des Trägers verbunden und somit unabhängig vom Speicherformat. Hierdurch ergibt sich der Vorteil, dass bspw. auch bei einer Konvertierung des Trägermediums in ein anderes Speicherformat die versteckten, zusätzlichen Daten nicht verloren gehen.

Ein Digitales Wasserzeichensystem besteht aus zwei wesentlichen Komponenten. Auf der einen Seite gibt es die Wasserzeicheneinbettung, die die zusätzlichen Daten mit dem Träger verbindet. Auf der anderen Seite befindet sich die Wasserzeichendetektierung, deren Aufgabe es ist, die eingebetteten Daten wieder auszulesen bzw. deren Vorhandensein zu prüfen. Ein derart modelliertes Wasserzeichensystem (siehe Abbildung 2.1) ähnelt dem aus der Informationstechnik bekannten Modell des Übertragungskanals mit Codierer und Decodierer. Der Wasserzeichenträger stellt den Kanal dar. Dieser Kanal kann Störungen unterlegen sein. Bei Wasserzeichenverfahren spricht man von beabsichtigten oder unbeabsichtigten Störungen bzw. Angriffen, die im Abschnitt 2.1.3 näher erläutert werden. Jede Veränderung, die der Träger erfährt, wirkt auch auf das mit ihm verbundene Wasserzeichen.

Die Literatur unterscheidet Wasserzeichenverfahren mit blinder und informierter Detektierung, bezeichnet auch als *oblivious* und *non-oblivious*. Letztere übertragen auf einem weiteren Kanal eine Seiteninformation zusätzlich zum markierten Träger. Bei den *oblivious* Verfahren

hingegen steht dem Detektor ausschließlich das mit dem Wasserzeichen versehene Trägersignal zur Verfügung. Ob Seiteninformationen zur Vereinfachung des Extraktions-/Detektionsprozesses eingesetzt werden können, hängt vom jeweiligen Anwendungsgebiet des Wasserzeichensystems ab (siehe Abschnitt 2.1.2).

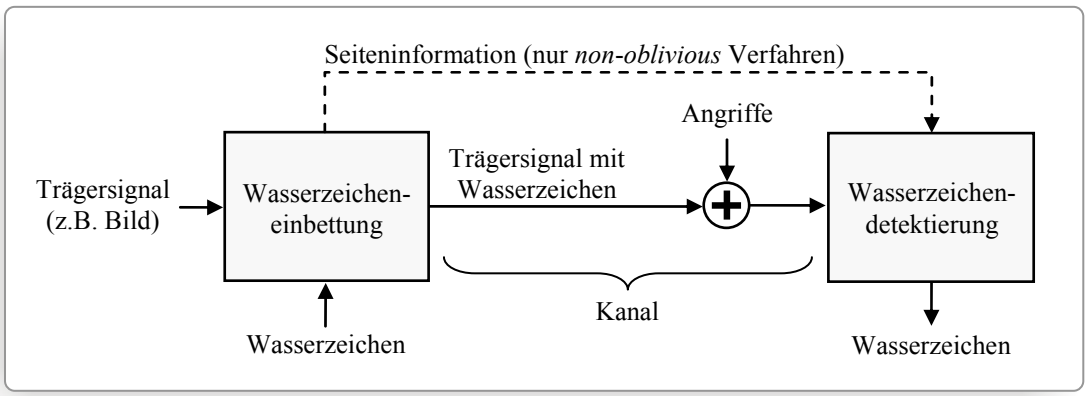


Abbildung 2.1 Modell eines Digitalen Wasserzeichensystems

2.1.1 Techniken der Wasserzeicheneinbettung

Notation und Begriffserläuterung

Auf unterster Ebene, in einer Art *Top-Down-Modell* für die Beschreibung von Wasserzeichensystemen (siehe Abbildung 2.2), befindet sich die Einbettungstechnik. Laut Literatur ergeben sich zwei unterschiedliche, in diesem Abschnitt beschriebene, Techniken [BB04]: das so genannte *Blind Embedding* und das *Direct Embedding*. Alle Untergruppierungen lassen sich diesen beiden Zweigen zuordnen.

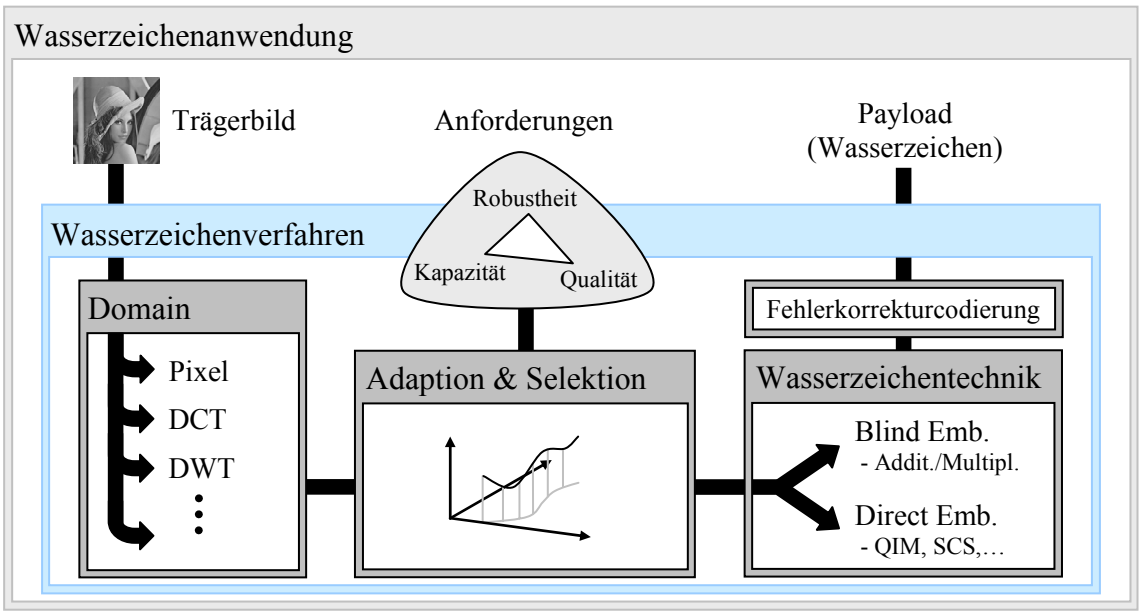


Abbildung 2.2 Top-Down-Modell eines Wasserzeichensystems

Die Wasserzeicheneinbettungstechnik wird in einer Domain (Transformationsraum) ausgeführt. Das bedeutet, das Wasserzeichen wird entweder im Pixelbereich auf einzelne Pixel oder Blöcke, auf Koeffizienten im Bereich der *Diskreten Cosinus*- oder *Wavelet-Transformation* oder auf Repräsentanten des Trägerbildes in einem beliebigen anderen Transformationsraum angewendet. Dabei können anwendungsabhängig die Eigenschaften der jeweiligen Domain genutzt werden. Transformationsdomains, wie bspw. die DCT-Domain[1], haben der Einbettung im Pixelbereich voraus, dass hier bereits eine Energieverdichtung und spektrale Zerlegung wirkt, wodurch weitere Berechnungen und Anpassungen vereinfacht werden.

Die im Träger einzubettende Nachricht $m := \{m_l \in \mathcal{M} : 1 \le l \le L\}$ der Länge L (*Payload*) können, abhängig von der jeweiligen Anwendung, bspw. eine *Signatur*, ein *Fingerprint*, ein Logobild und/oder sonstige Daten sein. Das Symbolalphabet von m sei in dieser Arbeit binär und somit $\mathcal{M} \in \{0,1\}$. Nach der Fehlerkorrekturcodierung wird daraus die bipolare Wasserzeichensequenz $w := \{w_k \in \pm 1 : 1 \le k \le K\}$ der Länge K. Wird keine Fehlerkorrektur angewendet, findet eine direkte Abbildung der Nachricht auf das Wasserzeichensignal statt: $w = 2 \cdot m - 1$ und $K = L$. Es sei definiert, dass je Einbettungsposition $p_k \in \mathbb{N}^+ : 1 \le p_k \le K$ ein Wasserzeichenbit im Träger $x := \{x_n \in \mathbb{R} : 1 \le n \le N\}$ der Länge N eingebettet wird. Das mit dem Wasserzeichen versehene Trägersignal sei dann $y := \{y_n \in \mathbb{R} : 1 \le n \le N\}$.

Ziel ist es, die Wasserzeichendaten so robust wie möglich gegenüber einer Vielzahl von Störungen im Trägersignal einzubetten, ohne den gewohnten Umgang mit dem Trägermedium zu beeinträchtigen. Das bedeutet in der Regel, dass die Veränderung des Trägersignals aufgrund der Wasserzeicheneinbettung nicht wahrnehmbar (störend) sein darf.

In Anpassung an die von der Wasserzeichenanwendung vorgegebenen Anforderungen hinsichtlich Robustheit, Kapazität[2] und Nicht-Wahrnehmbarkeit (Qualität) erfolgt vor dem Einbetten meist eine Adaption der Einbettungsstärke bzw. Selektion der Einbettungspositionen.

In den folgenden Unterabschnitten werden die grundsätzlichen Techniken der Einbettung erläutert und Beispiele für Wasserzeichenverfahren gegeben, die diese Techniken einsetzen.

Blind Embedding

Beim *Blind Embedding* wird das Wasserzeichen "ohne Rücksicht auf das Trägersignal" eingebettet. Das Wasserzeichen wird mit dem Träger gemischt[3]. Dabei gibt es die im Folgenden beschriebenen Untergruppierungen: *additive* und *multiplikative* Einbettung.

---

[1] Koeffizienten der *Diskreten Cosinus-Transformation* (DCT)

[2] Größe des Payload = Länge L der eingebetteten Nachricht m

[3] Der Begriff des Mischens zweier Signale ist auch aus der Nachrichtentechnik bekannt. Beim *Direct Embedding* hingegen spricht man nicht von einem Mischen, sondern von einem Ersetzen des Trägersignals.

Eingesetzt wird das Blind Embedding bspw. beim so genannten *Spread Spectrum Watermarking* [CMB+08], [HK99], [CKLS97] oder auch beim *Patchwork Watermarking* [BGML96], [KXYM07].

Additive Wasserzeicheneinbettung
Die am weitesten verbreitete Technik des Blind Embedding ist die additive Wasserzeicheneinbettung, wobei das bipolare Wasserzeichensignal w_k an der Position p_k, abhängig von einem die Einbettungsstärke bestimmenden Skalierungsfaktor γ, zum Träger x addiert wird:

$$y(p_k) = x(p_k) + \gamma \cdot w_k \tag{2.1}$$

Die additive Wasserzeicheneinbettung ist einfach zu implementieren und ausgiebig erforscht. Die Detektierung des Wasserzeichens kann mittels *Korrelation* erfolgen.

Die Einbettungsstärke kann speziell an Maskierungseigenschaften des menschlichen Sehens[4] oder an das Trägersignal angepasst sein. Der Skalierungsfaktor γ ist dann, wie in [SZTB98] oder [PZ98], eine Funktion des Trägers, $\gamma = f(x)$.

Multiplikative Wasserzeicheneinbettung
Bei der multiplikativen Einbettung wird das Wasserzeichen mit dem Träger multipliziert (Gleichung (2.2)) und dadurch direkt an die Intensität des Trägersignals angepasst.

$$y(p_k) = x(p_k) + \gamma \cdot w_k \cdot x(p_k) \tag{2.2}$$

Vor allem im Frequenzbereich eingesetzt, lässt sich auf diese Weise bereits generell ein visueller Frequenzmaskierungseffekt während der Einbettung erzielen. Im Pixelbereich angewendet, kann durch das multiplikative Einbetten der Effekt der Luminanzmaskierung ausgenutzt werden. Beispiele finden sich u.a. in [CKLS97] und [BBCP98].

Direct Embedding

Beim Direct Embedding wird das Trägersignal nicht mit dem Wasserzeichen gemischt, sondern so verändert, dass es im Sinne einer definierten Einbettungsregel eine vorgeschriebene Eigenschaft aufweist. Das Trägersignal wird dabei, abhängig vom Wasserzeichen, durch ein nach den Regeln des Einbettungsalgorithmus gebildetes Signal ersetzt (substituiert).

Wie in Abbildung 2.3 dargestellt, wird zuerst der Bereich aller ein gültiges Wasserzeichen beinhaltenden Signale R_w definiert. Dann wird das Trägersignal so verändert, dass es sich für den Wasserzeichendetektor in diesem Bereich befindet.

[4] Die drei Maskierungseigenschaften des menschlichen Sehsystems (engl. *Human Visual System*), Luminanz-, Frequenz- und Texturmaskierung, werden durch moderne Wasserzeichenverfahren ausgenutzt, um Daten so einzubetten, dass die dadurch verursachten Trägerbildveränderungen nicht wahrnehmbar sind.

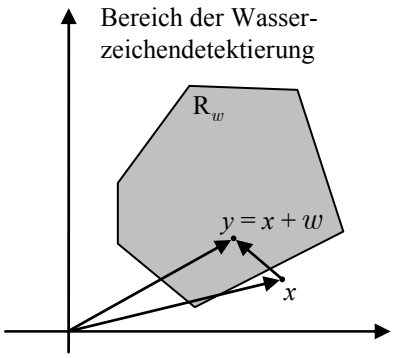


Abbildung 2.3 Direct Embedding: Bereich R_w für die Wasserzeichendetektierung (vgl. [BB04])

Das Trägersignal kann im Bereich R_w so platziert werden, dass entweder die wahrgenommene Trägersignalveränderung minimal oder die Robustheit der eingebetteten Daten maximal wird. Wenn während der Wasserzeicheneinbettung bekannt ist, wie das Trägersignal später durch Störungen verändert wird, kann derjenige Träger aus dem Bereich der gültigen markierten Signale gewählt werden, der nach der Störung vermutlich maximal robust sein wird. Diese Erweiterung wird als *Informed Coding* bezeichnet.

Ein Vorteil des Direct Embedding ist, dass es im Gegensatz zum Blind Embedding nie zu *Interferenzen* mit dem Träger kommt. Findet keine Veränderung des Trägersignals in Form einer Störung statt, können die Daten durch die Extraktion stets fehlerfrei ausgelesen werden.

Low-Bit Modulation

Die am längsten bekannte und einfachste Technik, die zum Direct Embedding zählt, ist die *Low-Bit-Modulation* (LBM). Dabei werden die niederwertigsten Bits eines Trägersignalwertes (Binärdarstellung) durch die an dieser Position einzubettenden Wasserzeichenbits ersetzt. Dieser unkomplizierte Ansatz zeichnet sich durch hohe Kapazität und geringe Bildverzerrungen aus. Die Robustheit ist jedoch sehr begrenzt. Man spricht bei LBM auch oft von fragilen (zerbrechlichen) Wasserzeichen.

Quantization Index Modulation

Weitaus komplexer ist der auch zur Gruppe der ersetzenden Wasserzeichentechniken zählende Ansatz von Chen und Wornell [CW01], bezeichnet als *Quantization Index Modulation* (QIM). Die Wasserzeichendaten werden eingebettet, indem ein Satz unterschiedlicher Vektorquantisierer $Q(\bullet) := \{Q_i(\bullet) : i = 0,1,2,...\}$ konstruiert und die Trägersignalwerte auf die Elemente (Quantisierungsgitterpunkte) dieser Quantisierer abgebildet werden. In Abbildung 2.4 besteht dieser Satz aus zwei Quantisierern, deren Gitterpunkte durch die Symbole × und o dargestellt sind. Zum Einbetten eines $w_k = -1$ wird der Trägersignalwert einem der Punkte × des Quantisierers $Q_{-1}(\bullet)$ angenähert, bei einem $w_k = +1$ einem der Punkte o von $Q_{+1}(\bullet)$.

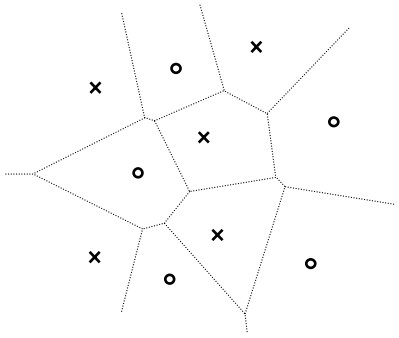


Abbildung 2.4 Gesamtquantisierungsgitter bestehend aus zwei Vektorquantisierern

Auf der Extraktionsseite wird der Trägersignalwert dem nächst dichtesten Punkt des Gesamtquantisierersatzes zugeordnet und das Wasserzeichenbit $\hat{w}_k$ nach Gleichung (2.3) bestimmt, wobei $Q_b^{-1}(\bullet)$ den zu $Q_b(\bullet)$ inversen Quantisierer bezeichnet.

$$\hat{w}_k = \arg\min_{b\in\pm 1} \left\| y(p_k) - Q_b^{-1}\left(Q_b\left(y(p_k)\right)\right) \right\| \tag{2.3}$$

Dither Modulation

Bei der *Dither Modulation* [CW01], [EG02] handelt es sich um eine Realisierung der QIM mit verringerter Komplexität, ähnlich der Beschreibung des *Scalar Costa Scheme* (SCS) von Eggers und Girod [ESG00].

Jedes Wasserzeichenbit w_k wird unabhängig von den restlichen Bits, separat mithilfe eines gleichmäßigen eindimensionalen Quantisierungsgitters $\Lambda \in \{\Lambda_{-1}, \Lambda_{+1}\}$ der Schrittweite Δ eingebettet (siehe Abbildung 2.5). Dieses Gitter besteht aus zwei in Abhängigkeit vom einzubettenden Bit w_k ausgewählten Untergittern $\Lambda_{w_k} := \{\Delta\mathbb{Z} + w_k \cdot \Delta/4 + d_m\}$. Der Wert d_m ist dabei ein schlüsselabhängiger Parameter[5] zur Erhöhung der Sicherheit des Systems. Der Wert d_m wird auch als Zitter-Vektor (engl. *dither*) bezeichnet. Er verändert sich von Bit zu Bit und verhindert so eine statistische Auffälligkeit, die ansonsten durch das Quantisieren entsteht.

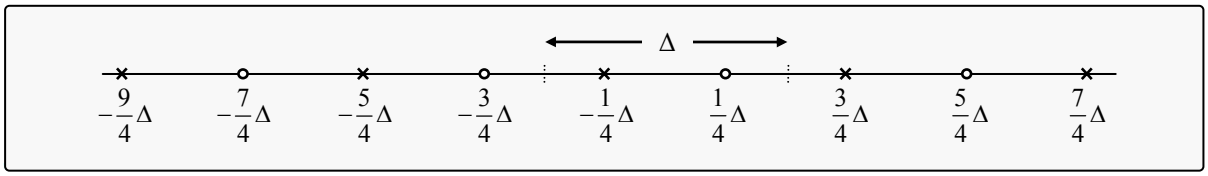


Abbildung 2.5 Gleichmäßige eindimensionale Quantisierung zur Wasserzeicheneinbettung

2.1.2 Anwendungsgebiete

In [CM02] geben Cox und Miller einen 50 Jahre währenden, geschichtlichen Überblick über die Nutzung von Wasserzeichenverfahren oder Systemen, die dem Einsatz von Wasserzeichen

[5] Der Einfachheit halber und ohne Einschränkung der Funktionalität sei im Verlauf dieser Arbeit $d_m = 0$.

nahe kommen. Vor allem im Audiobereich gab es schon seit längerem ein Interesse, die Eigentums- und Nutzungsrechte verkaufter oder ausgestrahlter Musik zu kontrollieren. Ein breit wissenschaftliches und kommerzielles Interesse kam allerdings erst innerhalb der letzten 15 Jahre auf, seitdem es mithilfe der Digitaltechnik möglich ist, Musik, Bilder und Videos rasend schnell, einfach und verlustfrei zu verbreiten bzw. zu erstellen und zu verändern.

Zu den vorrangigen Einsatzgebieten für Digitale Wasserzeichen zählen u.a. die Kontrolle des Copyright (*Copyright Control*), die Nutzungsnachverfolgung ausgestrahlter Medienangebote (*Broadcast Monitoring*), die Zugriffs- und Gerätekontrolle (*Device Control*), die Echtheits- bzw. Unversehrtheitsüberprüfung (*Authentifizierung*) von Multimediainhalten und diverse als *Legacy-Channel-Watermarking* bezeichnete Applikationen.

Copyright Control

Das wahrscheinlich kommerziell wichtigste Einsatzgebiet Digitaler Wasserzeichen ist die Kontrolle von Urheber- und Zugriffsrechten für Multimediaangebote. Vor allem die Musik- und Filmindustrie begründen ein großes Interesse daran, die Verbreitung und den Absatz ihrer Medien nachvollziehen und illegalen Kopierern das Handwerk legen zu wollen. So möchte man gerne den Weg, den ein in einer Videothek ausgeliehenes oder über ein *Video-on-Demand-System* beschafftes Video nimmt, nachverfolgen können. Kursiert ein nicht rechtmäßig kopiertes Bild, Video oder Musikstück anschließend für jedermann verfügbar im Internet, soll der Raubkopierer ausfindig gemacht und bestraft werden (*Traitor Tracing*). Zu diesem Zweck könnte ein eindeutiges Wasserzeichen (*Fingerprint*) in jede Kopie der verliehenen oder verkauften Daten eingefügt werden. Dieses Wasserzeichen müsste robust sein gegenüber Formatänderungen, dem Abschneiden von Randbereichen (Cropping), Skalierung und Angriffen, die den Fingerprint gezielt entfernen sollen.

Authentifizierung

Ein weiteres wichtiges Einsatzgebiet stellt die Authentifizierung digitaler Inhalte dar. Sollen Multimediadaten zwischen zwei Parteien ausgetauscht werden, ist es wichtig, die Unversehrtheit und den Ursprung dieser Daten nachweisen zu können. Wie aus der klassischen Kryptografie bekannt, könnte zu diesem Zweck mithilfe einer *Hash-Funktion* und *asymmetrischer Verschlüsselung* eine *Digitale Signatur* berechnet und parallel zum Multimediainhalt oder in dessen Metadaten gespeichert bzw. übertragen werden. Wird die Signatur jedoch als Wasserzeichen bspw. im Bild unsichtbar eingebettet, kann sie nicht verloren gehen. Dadurch ergibt sich ein Vorteil bei der Datenverwaltung.

Die Signaturgenerierung und das Einbetten können derart gestaltet werden, dass Operationen und Kompressionsformatänderungen zugelassen sind, die die Glaubwürdigkeit und Echtheit

nicht beeinflussen. So könnte ein geschütztes Bild z.B. ausgedruckt und von jedermann eingescannt oder abfotografiert und auf Echtheit/Ursprung überprüft werden. Eine detaillierte Darstellung derartiger Verfahren befindet sich in Kapitel 3.

Legacy-Channel

Wasserzeichenverfahren können auch genutzt werden, wenn die Funktionalität eines technischen Systems erweitert werden soll, ohne dabei die Kompatibilität zu bestehenden Geräten einzuschränken. Ältere Geräte (engl. *legacy devices*) könnten mit den alten Funktionen weiterverwendet werden. Neue Geräte könnten die zusätzlichen, als Wasserzeichen eingebetteten Daten und den erweiterten Funktionsumfang nutzen, wie bspw. in [SAL+89] und [PS98].

Auch das so genannte *Annotation-Watermarking* (Hinzufügung von Informationen) kann zur Gruppe der Legacy-Channel-Systeme gezählt werden. So stellen z.B. Van der Veen *et al*. in [VBH+01] oder Kirovski in [KM01] ein System vor, das zu einem Musikstück über das eingebettete Wasserzeichen Informationen zu Titel/Album/Interpret oder ein Verkaufsportal zur Verfügung stellt. Ein anderes System, beschrieben in [VS06], soll das Unterteilen von Bildern in Objekte sowie das Bezeichnen dieser erlauben. Eins ist allen Legacy-Channel-Systemen gemein, sie bringen zusätzliche Funktionalität. Ein Nutzer hat kein Interesse daran, das Wasserzeichen zu entfernen. Die Verfahren sind also keinen gezielten, bösartigen Angriffen ausgesetzt. Sie sollten dennoch robust sein gegenüber Standardoperationen wie z.B. Bildkompression, Skalierung oder Cropping.

2.1.3 Angriffe auf Digitale Wasserzeichen

Digitale Wasserzeichen werden, wie im vorherigen Abschnitt beschrieben, aus unterschiedlichen Gründen eingebettet. Während es Anwendungsgebiete gibt, wo keine Partei (Eigentümer, Nutzer, Angreifer, etc.) daran interessiert ist, die zusätzlich eingebetteten Daten unlesbar zu machen oder zu verändern, kann in anderen Fällen durchaus das Bestreben entstehen, die Detektierung des Wasserzeichens zu behindern.

Oft wird zwischen beabsichtigten und unbeabsichtigten Störungen unterschieden. Zu den letzteren zählen bei Bildern alle Arten von Bildbearbeitungsoperation, die den Bildinhalt nicht störend verändern. Beispielhaft dafür sind bildverbessernde Maßnahmen wie Schärfung, Helligkeits- und Kontraständerungen, aber auch Formattransformationen (JPEG-, JPEG2000-Kompression, …) oder geringfügige geometrische Bildveränderungen.

Im Gegensatz dazu können Angriffe mit Hintergrundwissen über die verwendete Wasserzeichentechnik oder grobe Veränderungen des Trägers als beabsichtigte Störungen verstanden werden. Eine klare Trennung ist jedoch nicht möglich. Vielmehr hängt die Differenzierung

zwischen beabsichtigten und unbeabsichtigten Störungen mit den Vorgaben der jeweiligen Wasserzeichenanwendung zusammen.

2.2 JPEG2000-Bildkompression

Der in dieser Arbeit vorgestellte Algorithmus zum Generieren und Einbetten eines Authentifizierungswasserzeichens ist in den Prozess einer JPEG2000-Einzelbildkompression integriert. Das Kernsystem dieser Kompression wird im Folgenden beschrieben. Die dabei zum Einsatz kommenden Komponenten sind in ähnlicher Form auch Bestandteil nahezu jeder anderen Bildkompression wie JPEG, SPHIT, H.264/AVC-Intra Frame Coding.

Mit JPEG2000 hat die *Joint Photographic Experts Group* (JPEG) einen sehr effektiven Kompressionsstandard für Einzelbilder entwickelt. Der neue Standard umfasst 13 Teile, die über den Basisstandard ISO/IEC 15444-1 (fertig gestellt: Jan. 2001) hinaus diverse Ergänzungen und Erweiterungen auf andere Anwendungsgebiete spezifizieren.

Der Basisstandard [ISO00] beschreibt nach einer Vorverarbeitung (engl. *pre-processing*) des zu komprimierenden Bildes die Transformation in den Wavelet-Bereich (DWT) sowie eine anschließende Quantisierung und Codierung der DWT-Koeffizienten (siehe Abbildung 2.6). Die Decodierung verläuft im Prinzip genau umgekehrt.

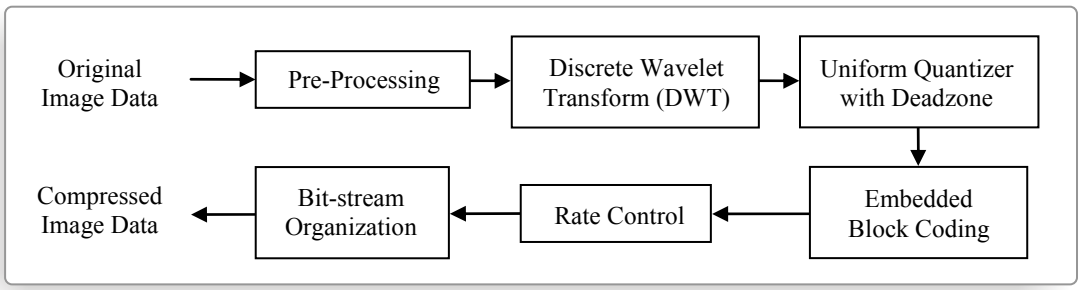


Abbildung 2.6 Ablauf der JPEG2000-Codierung nach dem Basisstandard (Quelle: [ISO00])

2.2.1 Pre-Processing

Zum Pre-Processing, der Vorverarbeitung eines zu codierenden Bildes, gehören, wie auch beim Vorgänger (JPEG-Kompression), ein so genannter *Level-Offset* und eine Farbraumtransformation. Dabei wird das Bild aus dem *RGB*-Farbraum in den YC_bC_r-Farbraum umgewandelt. Es wird in seine Luminanz- (Helligkeit) und Farbkomponenten (Differenz zu blau und rot) zerlegt und dadurch an das unterschiedlich gute visuelle Wahrnehmungsvermögen von Helligkeit und Farbe angepasst. Über die vom JPEG-Standard bekannte, verlustbehaftete Berechnungsvorschrift hinaus steht bei JPEG2000 auch eine verlustfreie *Reversible Color Transform* (RCT) zur Verfügung. Sie erlaubt, die Farbraumtransformation ganzzahliger Werte ohne Rundung durchzuführen.

Im Unterschied zum bekannten JPEG-Standard findet vor der Farbraumtransformation eine Aufteilung des Bildes in feste Bereiche (*Tiles*) statt. Jeder dieser Bereiche wird unabhängig und ggf. mit unterschiedlichen Parametern codiert. Dadurch kann der Speicheraufwand während des Komprimierens/Dekomprimierens begrenzt werden.

2.2.2 Diskrete Wavelet-Transformation

Den Kern des JPEG2000-Kompressionsprozesses stellt die Diskrete Wavelet-Transformation (DWT) dar. Sie ist eine Zeit-Frequenz-Transformation, die das Bild in einen Satz von Basisfunktionen in Form "kleiner Wellen" (engl. *wavelets*) zerlegt. Dabei handelt es sich im Prinzip um eine Faltung des Bildes mit einer Wavelet-Funktion. Diese Faltung wird zunächst am Original und danach in immer weiteren Schritten auf skalierte Abbilder des Originals vollzogen, um immer feiner differenzierte Transformationen zu erhalten. Auf diese Weise ergibt sich bereits durch die Wavelet-Transformation eine Mehrfachauflösung des Bildes, die im JPEG2000-Standard als inhärente Eigenschaft fortgeführt wird.

Die DWT lässt sich besonders effektiv in Form einer Zwei-Kanal-Filterbank realisieren, wobei das Signal in zwei aufeinanderfolgenden Schritten in jeweils zwei Komponenten, den Hoch- und den Tiefpassanteil, zerlegt wird. Das Bild passiert zuerst spaltenweise jeweils das Hoch- bzw. Tiefpassfilter und wird dann spaltenweise unterabgetastet. Anschließend wird es zeilenweise gefiltert und unterabgetastet (siehe Abbildung 2.7). Die sich ergebenden, in der Auflösung um die Hälfte reduzierten, Filterergebnisse (DWT-Koeffizienten) werden als DWT-Subbänder bezeichnet.

Die horizontalen Bildfrequenzen entsprechen dem Subband HL, die vertikalen dem Subband LH und die diagonalen Frequenzen erhalten das Kürzel HH. Das LL-Band der Zerlegung

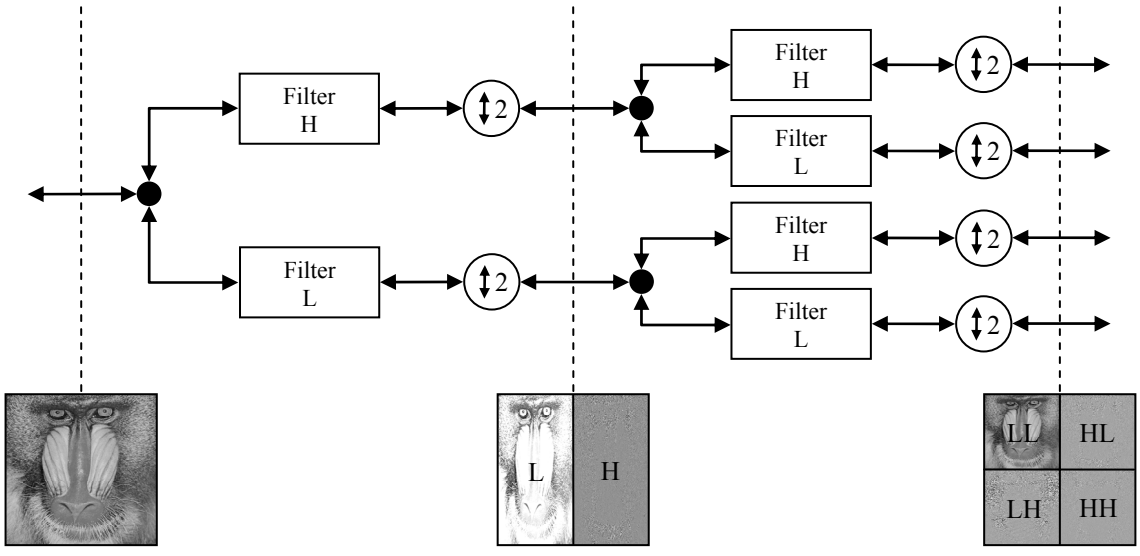


Abbildung 2.7 Zweidimensionale Wavelet-Zerlegung eines Bildes

repräsentiert das um den Faktor 2^Z verkleinerte Originalbild, wobei Z die Stufe der Mehrfachzerlegung der DWT bezeichnet. Das LL-Band ist nach der ersten Zerlegung Ausgangspunkt für die zweite Zerlegung, dessen LL-Band wiederum für die dritte usw. Insgesamt entstehen $3 \cdot Z + 1$ Subbänder, deren Koeffizienten das Originalbild repräsentieren. Die Anzahl der Koeffizienten aller Bänder ist identisch mit der Pixelanzahl des Bildes.

Die DWT kann entweder verlustfrei als reversibles Integer-5/3-Filter oder verlustbehaftet als nicht-reversibles Floating-Point-9/7-Filter eingesetzt werden. Über den Basisstandard hinaus ist auch der Einsatz anderer Wavelets erlaubt. Die Ergebnisse der Zerlegung können, wie in Abbildung 2.8 gezeigt, in einer Gesamtmatrix zusammengefasst werden.

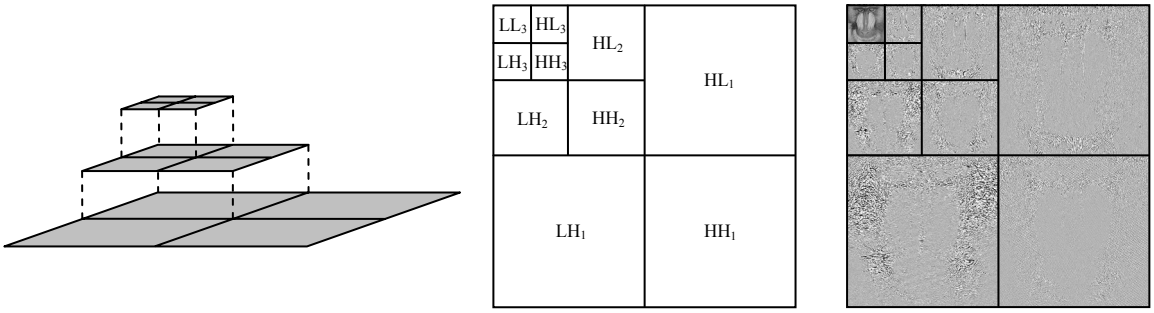


Abbildung 2.8 Zweidimensionale Wavelet-Zerlegung des Bildes Mandrill ($Z = 3$)

Mithilfe der Wavelet-Transformation wird das Bild in seine horizontalen, vertikalen und diagonalen Frequenzkomponenten zerlegt, die dann entsprechend ihrer visuellen Wichtigkeit weiterverarbeitet werden können. Durch die Transformation wird ein Großteil der Informationen auf einige wenige Koeffizienten konzentriert (Informationsverdichtung).

2.2.3 Quantisierung

Im Anschluss an die Wavelet-Transformation werden alle Koeffizienten einer gleichmäßigen, eindimensionalen Quantisierung (engl. *uniform scalar quantization*) zugeführt. Das heißt, die Koeffizienten werden durch eine feste Quantisierungsschrittweite Δ dividiert und auf den nächsten ganzzahligen Wert gerundet. Je Subband ist dabei eine separate Schrittweite zulässig. Die verlustbehaftete JPEG2000-Variante legt die Schrittweite derart fest, dass eine bestimmte Bildqualität für das komprimierte (rekonstruierte) Bild erreicht wird. Die finale Schrittweite eines jeden einzelnen Koeffizienten wird erst während der eingebetteten Blockcodierung[6] (siehe Abschnitt 2.2.4) geprägt.

Eine Besonderheit der bei JPEG2000 eingesetzten Quantisierung ist die so genannte Totzone (engl. *dead-zone*) um den Ursprung des Quantisierungsgitters (siehe Abbildung 2.9). Diese

[6] engl. *embedded block coding*

Totzone deckt den Bereich $[-\Delta;\Delta]$ ab und ist damit doppelt so breit wie die übrigen Quantisierungsintervalle. Über den Basisstandard von JPEG2000 hinaus sind auch weitere Formen der Quantisierung gestattet, bspw. mit variabler Totzonen-Breite oder der Einsatz einer *Trellis Coded Quantization* (TCQ).

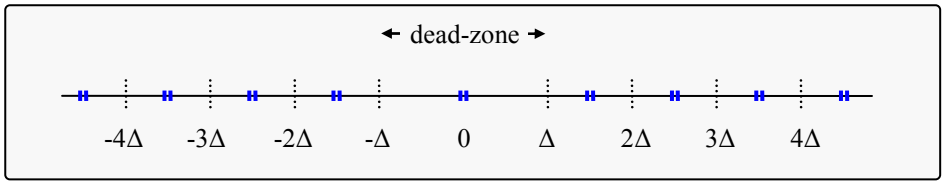


Abbildung 2.9 Eindimensionale Quantisierung mit Totzone

Für eine gegebene Schrittweite Δ generiert der Quantisierer $Q(\bullet)$ zu einem DWT-Koeffizienten a die vorzeichenbehaftete Integerzahl q entsprechend Gleichung (2.4). Die Rekonstruktion des Wertes durch den inversen Quantisierer $Q^{-1}(\bullet)$ folgt Gleichung (2.5).

$$q = Q(a) = \operatorname{sign}(a)\left\lfloor \frac{|a|}{\Delta} \right\rfloor \tag{2.4}$$

$$\hat{a} = Q^{-1}(q) = \begin{cases} 0 & q = 0 \\ \operatorname{sign}(q)\left(|q| + 0{,}5\right)\Delta & q \neq 0 \end{cases} \tag{2.5}$$

Eine weitere Besonderheit der Quantisierung bei JPEG2000 ist die Umsetzung in Form einer *sukzessiven Approximation* als Vorbereitung der Koeffizienten für die sich anschließende eingebettete Blockcodierung. Die Idee ist, dass jeder Koeffizient derart quantisiert wird, dass dessen Bitschreibweise eine progressive Verfeinerung erlaubt. Mit anderen Worten, der Decoder erstellt eine Approximation der zu rekonstruierenden Koeffizienten, sobald ein Teil des Bitstroms eintrifft. Je mehr Bits der Decoder zu einem Koeffizienten erhält, desto näher kommt die Approximation dem quantisierten Koeffizienten. Dazu wird die Schrittweite je Bitebene $j := \{ j \in \mathbb{N} : 0 \leq j < J \}$ als Vielfaches von $2^j\Delta$ gewählt, wobei $q = sq_0q_1q_2...q_{J-1}$ mit s = Vorzeichen, q_0 = MSB und q_{N-1} = LSB den quantisierten Wert vollständig beschreibt. Werden bei der Decodierung k Bitebenen wegelassen, so dass $q^{(k)} = sq_0q_1q_2...q_{J-1-k} = Q_k(a)$ einer Quantisierung mit der Schrittweite $2^k\Delta$ entspricht, ändert sich Gleichung (2.5) wie folgt:

$$\hat{a} = Q_k^{-1}\left(q^{(k)}\right) = \begin{cases} 0 & q^{(k)} = 0 \\ \operatorname{sign}\left(q^{(k)}\right)\left(\left|q^{(k)}\right| + 0{,}5\right)2^k\Delta & q^{(k)} \neq 0 \end{cases} \tag{2.6}$$

2.2.4 Eingebettete Blockcodierung

Die einzelnen Bits der Transformationskoeffizienten werden nun so organisiert, dass ein gezielter Zugriff auf einzelne Bildbereiche eine bestimmte Bildauflösung oder Bildqualität ermöglicht, ohne dass der gesamte Bitstrom decodiert werden muss. Dadurch wird erreicht, dass ein Bild einmal codiert und dann vielfach für unterschiedliche Anwendungen, Geräteklassen und Übertragungskapazitäten genutzt werden kann. Es werden dabei nur die jeweils benötigten Bits aus der codierten Struktur herausgelöst und übertragen bzw. decodiert.

Die Subbänder werden in Codeblöcke fester Größe (z.B. 64x64) unterteilt und fortan unabhängig behandelt. Ziel der eingebetteten Blockcodierung, bezeichnet als *Embedded Block Coding with Optimal Truncation* (EBCOT), ist es, den Code, der durch die einzelnen Codeblöcke entsteht, so zu begrenzen, dass in der Summe eine vorher festgelegte Datenrate entsteht. Die Codebegrenzung erfolgt dabei durch das Weglassen einzelner Bits der zuvor beschriebenen progressiven Bitschreibweise der Koeffizienten. Als Kriterium für die Auswahl der Bits gilt die Minimierung der dadurch entstehenden Gesamtverzerrung des Codierungsvorganges. Diese optimierte Codebegrenzung erfolgt also im eigentlichen Sinne erst nach dem kompletten Codiervorgang des jeweiligen Tile samt *Arithmetischer Codierung* [Str05]. Für eine detaillierte Beschreibung des EBCOT-Prozesses sei auf Taubmans Dokumentation [TM02] der bekannten Kakadu-Softwareimplementierung verwiesen.

2.2.5 Decodierung und Vorteile gegenüber der JPEG-Kompression

Die Decodierung eines mittels JPEG2000 komprimierten Bildes verläuft im Wesentlichen genau umgekehrt zur Encodierung. Neu ist bei JPEG2000 jedoch, dass einzelne Komponenten übersprungen werden können. So ist es bspw. möglich, ohne den gesamten Bitstrom decodieren zu müssen, nur einzelne Teile herauszulösen und nur eine verkleinerte oder qualitativ eingeschränkte Version des Bildes bzw. nur einen Teilbereich zu rekonstruieren. Diese sich von Anfang bis Ende durch den JPEG2000-Standard erstreckende sehr effektive Skalierungseigenschaft ist der größte Unterschied zum Vorgänger JPEG. Sie schließt auch das so genannte Codieren einer Region erhöhten Interesses[7] ein, wobei einzelne Bildbereiche weniger stark komprimiert werden können und in höherer Qualität verbleiben als andere.

Die Kompressionsleistung ist im Gegensatz zu JPEG nur geringfügig höher. Erst bei sehr großen Bildern spielt JPEG2000 durch die größere Länge der Wavelet-Filter gegenüber den relativ kleinen 8x8 Blöcken der DCT von JPEG seine Vorteile aus. Es entstehen bei hohen Kompressionsraten nicht die von JPEG bekannten Blockartefakte. Stattdessen tendieren die

[7] engl. *region of interest* (ROI)

Bilder zu Unschärfe und Schatten an Objekträndern. Letzten Endes ist die subjektive Bildqualität von JPEG2000 in den meisten Fällen besser. Deutliche Vorteile ergeben sich außerdem durch die Möglichkeit der verlustfreien Kompression sowie *Transcodierung* JPEG2000-codierter Bilder ohne inkrementelle Rundungsfehler.

2.2.6 JPSEC

Die JPEG2000-Kompression umfasst 13 Abschnitte, die über den Basisstandard ISO/IEC 15444-1 hinaus diverse Erweiterungen auf andere Anwendungsgebiete spezifizieren. Teil 8 beschreibt dabei Werkzeuge und Lösungen, um JPEG2000-codierten Bildern wichtige neue Eigenschaften wie Zugriffkontrolle, Authentizität, Integrität[8] und Urheberschutz zu verleihen. Dieser Teilstandard trägt den Namen JPSEC und wurde im März 2006 fertig gestellt.

JPSEC ist im Grunde eine Definition von 10 Werkzeugen [ISO06], die sich unterschiedlicher Techniken wie Verschlüsselung, Schlüsselmanagement, Signaturgenerierung und -verifikation sowie *Scrambling*[9] und Wasserzeichen bedienen. Dabei sind die einzelnen Schutzmechanismen direkt an das JPEG2000-Kompressionsformat angepasst, so dass sich ein um JPSEC erweiterter Bitstrom wie gewöhnliche nach dem Basisstandard codierte Daten verhalten. Das bedeutet, trotz der Erweiterung sind die JPEG2000-Eigenschaften wie Skalierung der Auflösung oder der direkte Zugriff auf Teilkomponenten des Bildes auch weiterhin gegeben. Auf diese Weise wird die so genannte *Ende-zu-Ende-Sicherheit* eines geschützten Bildes durch JPEG2000-Transcoding (Umstrukturierung des Bitstroms) nicht beeinflusst.

2.3 Kanalcodierung zur Fehlerkorrektur

Gegenstand dieses Abschnitts ist die Einführung in die *Faltungscodierung*, die in dieser Arbeit zur Korrektur von Übertragungsfehlern verwendet wird. Da der zur Decodierung von Faltungscodes eingesetzte *Viterbi-Decoder* im Abschnitt 6.3 zur Erlangung neuer Funktionalität gezielt verändert wird, erfolgt hier eine detaillierte Erläuterung dessen Funktionsweise.

2.3.1 Faltungscodierung

Die Faltungscodierung ist ein Fehlerschutzmechanismus der Kategorie *Forward Error Correction* (FEC). Dabei handelt es sich um eine Einmaldatenübertragung, bei der der Sender der zu übertragenden Nachricht redundante Daten hinzufügt. Diese zusätzlichen Daten erlauben es dem Empfänger, Fehler bei der Übertragung zu erkennen und zu korrigieren, ohne vom Sender erneut Daten anfordern zu müssen.

[8] Eine Beschreibung der JPSEC-Integritätsprüfung (Bildauthentifizierung) erfolgt in Abschnitt 3.1.2.

[9] Scrambling = Schlüsselabhängige pseudozufällige Vertauschung der Reihenfolge der Werte einer Sequenz.

Bei der Faltungscodierung wird die zu übertragende Information im Sinne einer mathematischen Faltung über mehrere Stellen des Codewortes verteilt. Dabei bildet der Faltungscodierer im Regelfall k Informationsbits auf ein n Bit langes Codewort ab. Zudem sind aufeinander folgende Codewörter voneinander abhängig, wodurch der Faltungscodierer im Gegensatz zu *Blockcodes* ein so genanntes "Gedächtnis" besitzt.

Das Blockdiagramm einer Faltungscodierung ähnelt dem eines *FIR-Filters*. Ein FIR-Filter lässt sich als zeitdiskretes Schieberegister realisieren, bei dem nach jedem einzelnen der r Register ein Abgriff besteht, der jeweils mit einem Koeffizienten bewertet wird. Die bewerteten Abgriffe werden aufsummiert und bilden das Ausgangssignal.

Die Anzahl der Schieberegisterstellen bestimmt bei der Faltungscodierung die Länge des Gedächtnisses. Diese wird auch als *Constraint Length* bezeichnet. Das aktuelle Ausgangssignal c hängt somit von der Vergangenheit des Eingangssignals m ab. Beim klassischen Faltungsencoder existiert jedoch im Gegensatz zum FIR-Filter keine Bewertung der Abgriffe, diese werden ausschließlich miteinander XOR-verknüpft. Die Abgriffe sind so gewählt, dass sich eine möglichst große *Distanz* zwischen den Codewörtern ergibt.

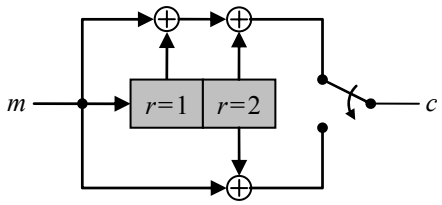


Abbildung 2.10 Blockdiagramm eines Faltungsencoders mit der Constraint Length 3 ($R_c = 1/2$)

Abbildung 2.10 zeigt den in allen Simulationen dieser Arbeit eingesetzten Faltungsencoder. Jedes Eingangsbit führt hier pro Schiebetakt zu zwei Bits am geschalteten Ausgang, wodurch die Coderate $R_c = 1/2$ beträgt.

Zustandsdiagramm

Bei einem Faltungsencoder handelt es sich im Grunde um eine Zustandsmaschine[10]. Die Zustände sind dabei die im Schieberegister gespeicherten Nachrichtenbits. Für das Beispiel mit $r = 2$ Registern können sich $2^r = 4$ Zustände Ⓘ, Ⓘⓘ, Ⓘⓘⓘ, Ⓘⓥ ergeben. Zur Veranschaulichung ist in Abbildung 2.11 ein Zustandsdiagramm dargestellt. Es zeigt, abhängig vom vorherigen Zustand des Systems (Registerbelegung), welcher Zustand im Folgeschritt erreicht wird bzw. wie die Ausgangsbits lauten, wenn ein neues Bit am Eingang anliegt. Die Werte-

[10] engl. *Finite-Sate Maschine*

tupel an den gerichteten Pfeilen des Zustandsdiagramms beschreiben das Signal in der Form Eingangssignal/Ausgangssignal.

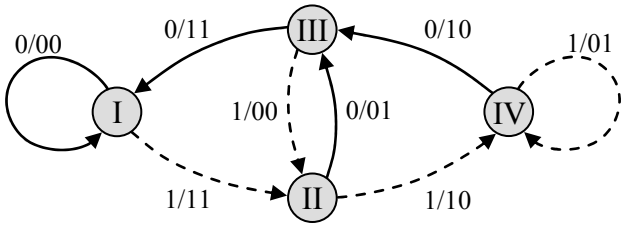


Abbildung 2.11 Zustandsdiagramm des Faltungsencoders aus Abbildung 2.10

Trellis-Diagramm

Um die zeitliche Abfolge der Zustände in der Zustandsmaschine besser demonstrieren zu können und den Codiervorgang zu verdeutlichen, eignet sich die Darstellung in einem Trellis-Diagramm. Es besteht aus einzelnen Segmenten, die für den entsprechenden Faltungsencoder alle möglichen Übergänge von einem Zustand in den Folgezustand widerspiegeln.

Abbildung 2.12 zeigt das Trellis für den in Abbildung 2.10 beschriebenen Encoder, wobei horizontal die zeitliche Abfolge, also der jeweilige Codierschritt, zu entnehmen ist. Vertikal sind die einzelnen Zustände vermerkt.

Das System startet zum Zeitpunkt $i = 0$ im Zustand I mit der Registerbelegung (00). Liegt am Eingang als Informationsbit eine "0" an, so führt die durchgezogene Linie im Trellis-Diagramm zum nächsten gültigen Zustand des Systems. Liegt eine "1" an, dann ist es die gestrichelte Linie, die auf den entsprechenden Folgezustand verweist. Die an den jeweiligen Pfaden vermerkten Wertetupel stellen wie beim Zustandsdiagramm die Codebits des Ein- und Ausgangssignals dar. Zum Schluss der Codierung, nachdem alle *k* Bits des Eingangssignals in das Schieberegister des Encoders eingelaufen sind, folgen so genannte Tail-Bits. Sie sorgen dafür, dass der Endzustand (Inhalt der Register) des Faltungsencoders wieder (00) und damit gleich dem Anfangszustand ist.

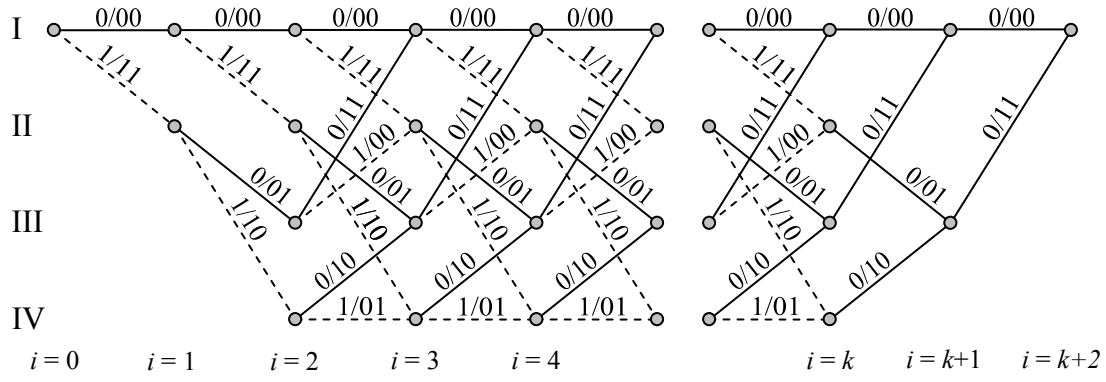


Abbildung 2.12 Trellis-Diagramm des Faltungscoders aus Abbildung 2.10

Zu jeder denkbaren Eingangssignalfolge ergibt sich genau ein Pfad durch das Trellis-Diagramm. Zum Beispiel führt die kurze Folge m = 0 1 1 0 1 0 1 1 der Länge k = 8 zur codierten Ausgangssequenz c = 00 11 10 10 00 01 00 10 10 11 der Länge n = 20.

2.3.2 Viterbi-Decoder

Der Viterbi-Algorithmus, 1967 erstmals vorgestellt von Andrew J. Viterbi [Vit67], ist das meistgenutzte Decodierungsschema für Faltungscodes. Es beruht auf dem *Maximum-Likelihood-Prinzip*, wonach der Decoder zu einer codierten Sequenz $\hat{c}$ eine, nach den Regeln des während der Faltungscodierung verwendeten Trellis gültige, Sequenz $\tilde{c}$ erzeugt. Die Sequenz $\tilde{c}$ wird so gewählt, dass sie der Sequenz $\hat{c}$ mit der höchsten Wahrscheinlichkeit entspricht. Die zur ermittelten Sequenz $\tilde{c}$ gehörige, decodierte Nachricht sei $\hat{m}$.

Grundsätzlich vollzieht der Viterbi-Decoder zur Bestimmung der wahrscheinlichsten Codesequenz zwei Operationen. In einem Vorwärtsschritt berechnet der Algorithmus für jeden Knoten des Trellis (Zustand zum Zeitpunkt i) die Wahrscheinlichkeiten der Pfade, die zu diesem Knoten führen. Im zweiten Schritt, einem Rückwärtsalgorithmus, wird für den wahrscheinlichsten Pfad zu jedem Knotenpunkt der zu ihm verweisende Vorgängerknoten ermittelt.

Die Wahrscheinlichkeiten der Pfade zu einem bestimmten Knoten ergeben sich aus dem jeweiligen Grad der Übereinstimmung der zu diesen Pfaden gehörigen Sequenzen mit der zu decodierenden Sequenz $\hat{c}$. Dieser Übereinstimmungsgrad wird im Viterbi-Decoder für jeden Knotenpunkt in einer so genannten Metrik festgehalten. Die Entscheidung, welches der wahrscheinlichste Pfad von einer Pfadverzweigung zu einem späteren Knoten ist, wird erst getroffen, wenn die unterschiedlichen Pfade in einem anderen Knoten zusammentreffen. Für das Trellis-Diagramm aus Abbildung 2.12 ist dies erst nach mindestens drei Codierschritten, also der einfachen *Constraint Length*, der Fall. Im schlimmsten Fall treffen zwei Pfade jedoch erst nach der Decodierung der kompletten Sequenz wieder zusammen. Meist genügt jedoch eine Betrachtung von weniger als der vier- bis fünffachen *Constraint Length*. Mit dieser fensterähnlichen Trellis-Decodierung wird vermieden, dass der Speicherbedarf exponentiell mit der Länge des Informationssignals wächst.

Die Metrik $M := \left\{ M_{ij} \in \mathbb{R} : 1 \le i \le \text{Fensterlänge}, 1 \le j \le 2^r \right\}$ beschreibt für die Knoten des Trellis zu jedem Betrachtungszeitpunkt innerhalb der Fensterlänge die Übereinstimmung der Sequenzen der zu diesen Knoten führenden Pfade mit der zu decodierenden Sequenz. Diese Übereinstimmung wird aus der Distanz der Eingangsbits eines jeden Decodierschritts zu den die gültigen Pfade beschreibenden Bits berechnet. Ein Pfad ist also am wahrscheinlichsten, wenn seine Bitsequenz die geringste Distanz zur Eingangsfolge aufweist. Dieser Pfad ist dann zum Entscheidungszeitpunkt der "überlebende" Pfad.

Für die codierte Beispielsequenz c = 00 11 10 10 00 01 00 10 10 11 wäre die Mindestdistanz im ersten Decodierschritt zum Knoten $I_{i=1}$ durch $M_{11}^{\min} = 0$ gegeben. Zum Knoten $II_{i=1}$ ist $M_{12}^{\min} = 2$. Die Distanzen $M_{13}^{\min}$ und $M_{14}^{\min}$ sind nicht definiert, da diese Zustände zum Zeitpunkt $i = 1$ nicht eingenommen werden können. Im zweiten Decodierschritt ergäben sich aufsummiert $M_{21}^{\min} = 2$, $M_{22}^{\min} = 0$, $M_{23}^{\min} = 3$ und $M_{24}^{\min} = 3$. Im dritten Decodierschritt wären $M_{31}^{\min} = 3$, $M_{32}^{\min} = 3$, $M_{33}^{\min} = 2$ und $M_{34}^{\min} = 0$. Mit einer Fensterlänge von 3 würde die Entscheidung nach diesem dritten Schritt auf Knoten $IV_{i=3}$ als Knoten mit der geringsten Metrik fallen. Der Rückwärtsalgorithmus würde dann den Pfad $IV_{i=3} \rightarrow II_{i=2} \rightarrow I_{i=1} \rightarrow I_{i=0}$ und somit bis dahin die Sequenz $\tilde{c}$ = 00 11 10 decodieren.

Treten innerhalb der codierten Sequenz Substitutionsfehler auf (binär: 0→1 bzw. 1→0), so kann zu dieser fehlerbehafteten Sequenz $\overline{c}$ stets ein gültiger Trellis-Pfad gefunden werden. Übersteigt die Anzahl der gleichzeitig innerhalb der *Constraint Length* gebündelt auftretenden Fehler jedoch die Korrekturfähigkeit des Decoders, kann dieser die Nachricht nicht fehlerfrei decodieren. In demjenigen Knoten, wo sich der fehlerhafte und der zur fehlerfreien Sequenz gehörende Trellis-Pfad treffen, ist die Korrekturfähigkeit für Folgebits wiederhergestellt.

Der Viterbi-Decoder erlaubt den Umgang mit "weichen", nicht binär vorentschiedenen Eingangsdatenfolgen (engl. *soft-input*). Dann ergibt sich die Metrik nicht wie im binären Fall aus der *Hamming-Distanz*, sondern aus der *Euklidischen Distanz*.

Kapitel 3

JPEG2000-basierte semi-fragile Bildauthentifizierung

In diesem Kapitel wird ein neues Verfahren zur Überprüfung der Echtheit von Bildern mithilfe Digitaler Wasserzeichen entwickelt. Dafür werden in den nachfolgenden Abschnitten zunächst einige Begriffe und Anforderungen formuliert und anschließend die neuen Algorithmen zur Generierung und zum Einbetten/Überprüfen des Wasserzeichens detailliert beschrieben. Danach wird die Funktions- und Leistungsfähigkeit unter Beweis gestellt, woraufhin das entwickelte Verfahren mit bekannten Verfahren anderer Autoren verglichen wird.

3.1 Überprüfung der Echtheit von Bildern

Der Wandel von der klassischen analogen zur digitalen Fotografie führte in den vergangenen Jahren zu zahlreichen vorteilhaften Neuerungen. Bilder und Videos lassen sich einfacher und mit höherer Qualität erstellen als je zuvor. Verbreitung und Vervielfältigung digitaler Inhalte erfolgen ohne Qualitätsverlust. Die Nachbearbeitung erfordert keine Spezialkenntnisse oder aufwändige Ausrüstung und ist trotzdem bildpunktgenau. Jedoch können gerade diese Neuerungen auch unangenehme Nachteile mit sich bringen. So können digitale Bilder und Videos mit einfachen Mitteln unnachweisbar inhaltlich verändert werden. Abbildung 3.1 gibt dazu ein simples Beispiel.

Abbildung 3.1 Beispiel einer den Bildinhalt betreffenden Manipulation. Originalbild (links), Fälschung (mittig), Differenzbild mit erhöhtem Kontrast (rechts)

In den Printmedien wird jedes Jahr mindestens eine spektakuläre Bildfälschung aufgedeckt. Das Repertoire reicht von der Retuschierung kleinerer Schönheitsfehler oder Pleiten bis hin zur Dramatisierung von Kriegsberichterstattungen oder politischen Kampagnen. Der Einsatz von Bildern und Videos als Beweismittel vor Gericht ist daher allgemein schwierig.

Um nachträgliche Bildmanipulationen erkennen zu können, gibt es zwei Möglichkeiten. Die erste, im Zeitalter der Digitalfotografie auch als *Digitale Forensik* bezeichnet, bemüht im Zweifelsfall (vor Gericht) speziell geschulte Gutachter, um Spuren einer eventuellen Bildbearbeitung aufzudecken. So erzeugt jede Kamera einen "Fingerabdruck" im aufgenommenen Bild (z.B. spezielle Rauschmuster [LFG06], [Far06], richtungsabhängige Farbsäume [JF06] oder Objektschatten [JF05]). Werden Objekte im Bild gelöscht oder aus unterschiedlichen Bildern gemischt, können Auffälligkeiten erkannt werden[11]. Eine andere Möglichkeit der Echtheitssicherung besteht darin, direkt nach der Bildaufnahme während des Kompressionsprozesses ein Digitales Wasserzeichen in das Datenmaterial einzufügen. Die komprimierten und gegen Fälschung geschützten Daten können dann im Speicher der Kamera abgelegt werden. Bei einem späteren Zugriff kann die Echtheit (Authentizität) auch ohne den Einsatz von Forensik-Experten, durch Überprüfung des Wasserzeichens, ermittelt werden.

3.1.1 Begriffserläuterungen und Anforderungen

Authentifizierung

Die Überprüfung der Echtheit von Daten (Verifizierung) ist eine aus der Kryptografie bekannte Technik [Buc03]. Dort wird, um die Unversehrtheit (Integrität) zu beweisen, aus den Daten mithilfe einer *kollisionsresistenten Hash-Funktion* (z.B. SHA-2[12], MD5,…) eine Prüfsumme (Hash-Wert) berechnet. Diese wird mit einem geheimen Schlüssel zu einer Signatur verschlüsselt (signiert). Zur Überprüfung wird die Signatur wieder entschlüsselt und mit dem erneut aus den zu verifizierenden Daten gebildeten Hash-Wert verglichen. Verbindet man die Verifizierung mit einem Identitätsnachweis (Zuweisung des Urhebers bzw. der Quelle), so handelt es sich um eine Authentifizierung der Daten.

Semi-Fragilität

Kollisionsresistente Hash-Funktionen haben die Eigenschaft, dass die generierte Prüfsumme bei auch nur der geringsten Veränderung des Ausgangssignals (z.B. Veränderung eines einzigen Bits) einen anderen Wert annimmt. Sie sind in diesem Sinne vollkommen fragil, was für

[11] Bekannte Forschungsgruppen um Prof. H. Farid am Dartmouth College (USA) und Prof. J. Fridrich an der Binghamton University (USA) entwickeln Programme zur automatischen Erkennung von Bildfälschungen.

[12] SHA-2 ist der Nachfolger des *Secure Hash Algorithm* (SHA-1) und umfasst die Gruppe der Hash-Funktionen SHA-224, SHA-256, SHA-384 und SHA-512.

die Echtheitsüberprüfung von Multimediadaten nicht wünschenswert erscheint. Schließlich sollen bestimmte Bildveränderungen, wie etwa die Transformation in ein anderes Kompressionsformat, erlaubt sein, solange die bildinhaltliche Aussage nicht gestört oder die Qualität durch Kompressionsartefakte nicht zu stark herabgesetzt wird. Auch eine Änderung der Bildgröße oder die nachträgliche Aufhellung/Abdunklung des Bildes sollen gestattet sein, solange dadurch keine wesentlichen, inhaltlichen Informationen verloren gehen. Andererseits sollen jedoch Manipulationen, wie etwa das Löschen oder Einfügen von Bildobjekten bzw. das Verändern des Bildinhaltes, erkannt und angezeigt werden.

Die *Digitale Forensik* ist bisher nicht dazu geeignet, erlaubte und unerlaubte Bildnachbearbeitungen sicher zu unterscheiden. Mit dem Ziel, erlaubte Bildnachbearbeitungen zuzulassen, jedoch den Bildinhalt verändernde Manipulationen zu detektieren, wurden semi-fragile Authentifizierungsverfahren auf der Basis Digitaler Wasserzeichen entwickelt. Einen Überblick dieser Verfahren geben [ESC+04], [ZST04], [RD02] und [SC05].

Bildabhängigkeit

Es existieren Verfahren, die zur Authentifizierung ein bekanntes, trägerunabhängiges Muster oder Logo in das Bild einbetten und bei korrekter Detektierung den Träger als echt ausweisen. Dieser Ansatz ist jedoch unsicher, da ein Angreifer das Bild manipulieren und das Logo erneut einbetten kann, ohne einen Alarm auszulösen. Der einzig sichere Weg ist es, aus dem Bildinhalt des zu schützenden Trägers einen Hash-Wert zu generieren, zu verschlüsseln und im Bild unterzubringen. Das Wasserzeichen muss also vom Bild direkt abhängig sein.

Einmaligkeit - Nachweis der Bildherkunft

Zusätzlich zur Signatur für die Erkennung unerlaubter Bildmanipulationen kann das eingebettete Wasserzeichen auch noch weitere Informationen enthalten. So können bspw. auch das Datum, die Uhrzeit sowie der Ort der Kameraaufnahme (z.B. GPS-Daten) fest in das Bild integriert werden. Auf diese Weise lässt sich verhindern, dass mit derselben Kamera eine Szene zu einem anderen Zeitpunkt, an einem anderen Schauplatz nachgestellt wird. Auch andere Maßnahmen, die in der Summe ein sicheres Gesamtsystem ergeben, sind denkbar. So meldete der Kamerahersteller Canon ein Patent[13] an, direkt während der Aufnahme in der Kamera einen Scan der menschlichen Iris des Fotografen als Wasserzeichen im Bild einzubetten. Auf diese Weise soll der Fotograf einer Szene eindeutig identifiziert werden.

Für die praktische Anwendung des Authentifizierungsverfahrens in einem mobilen Gerät ist es wichtig, dass das aufgenommene Bild die Kamera nicht ungeschützt verlässt. Die Echt-

[13] Canon's Iris Registration - Patent No.: 2008/0025574 A1

heitssicherung sollte also direkt nach der Aufnahme, während der Komprimierung in ein und demselben Gerät stattfinden [Fri93]. Es ist zwar auch möglich, ein Bild ungeschützt aufzunehmen und es erst anschließend dem Authentifizierungsalgorithmus (z.B. im PC) zuzuführen, jedoch wäre das ungesicherte Bild in der Zwischenzeit willkürlich bildinhaltlichen Manipulationen durch den Nutzer unnachweislich ausgesetzt.

Öffentliche Verifizierung

Die Sicherheit einiger bekannter Systeme beruht hauptsächlich auf einer Geheimhaltung des verwendeten Verfahrens oder der für das Wasserzeichen benutzten Trägersignalpositionen. Es ist jedoch wichtig, dass die Echtheit eines Bildes zu jeder Zeit und von jedermann überprüft werden kann. Das heißt, dass das verwendete Authentifizierungssystem ausführlich beschrieben und frei zugänglich sein muss. Dieser Gedanke folgt dem *Prinzip von Kerkhoff*, wonach die Sicherheit eines kryptografischen Verfahrens nicht von der Geheimhaltung der Ver- bzw. Entschlüsselungsfunktion abhängt. Für ein öffentlich zugängliches Verifizierungssystem kann zur Ver- und Entschlüsselung des Hash-Wertes ein so genannter *privater* und ein *öffentlicher Schlüssel* verwendet werden (*asymmetrische Verschlüsselung* [Buc03]). Zur Überprüfung der Echtheit ist ausschließlich der öffentliche Schlüssel notwendig, der keine Rückschlüsse auf den privaten, zur Verschlüsselung benutzten Schlüssel zulässt. Jeder kann die Echtheit überprüfen, aber niemand kann ohne Kenntnis des privaten Schlüssels eine gültige Signatur erzeugen. Der private Schlüssel und der Verschlüsselungsalgorithmus müssen sich folglich in einer sicheren Umgebung befinden, bspw. fest integriert im Chip der Kamera. Auch der Fotograf darf keinen Zugriff auf diese Daten haben.

Es ergeben sich zusammenfassend folgende Anforderungen an eine sichere Bildauthentifizierung. Sie werden alle von dem in dieser Arbeit entwickelten Verfahren erfüllt.

- **Integrität** (Algorithmus soll bösartige Bildmanipulationen erkennen)
- **Einbettung** (feste Verknüpfung) der Signatur in das zu schützende Bild
- **Robustheit** der Signatur gegenüber nicht bösartigen Bildveränderungen
- **Abhängigkeit** der eingebetteten Signatur von dem zu schützenden Bildinhalt
- **Unsichtbarkeit** der eingebetteten Signatur (keine sichtbaren Qualitätseinbußen)
- ***Obliviousness*** (Verifizierung sollte kein Originalbild benötigen)
- **Öffentliche Verifizierung** (Überprüfung durch jedermann)
- **Einmaligkeit** (gleichzeitige Sicherung von Ort, Zeit, Datum der Aufnahme)
- **Verhinderung des Zugriffs** auf den privaten Schlüssel und auf das Originalbild

Neben diesen vor allem sicherheitstechnisch wichtigen Anforderungen an ein Wasserzeichensystem zur Authentifizierung von Bildern ist auch von Bedeutung, dass dieses System effizient und an die Eigenschaften moderner Kompressionsverfahren angepasst arbeitet. Das be-

deutet, wenn die Signaturgenerierung und -einbettung in einem mobilen Gerät, z.B. einer Kamera oder einem Handy, direkt nach der Bildaufnahme stattfinden soll, darf die Berechnung nicht zu lange dauern. Außerdem dürfen die speziellen Funktionen des Multimediaformates nicht behindert werden, etwa in JPEG2000 der direkte Zugriff auf einzelne Teilbildbereiche, Auflösungsstufen oder Qualitätsprofile. Daher ist es sinnvoll, die Signaturgenerierung und das Einbetten des Wasserzeichens, wie in Abschnitt 3.2.1 beschrieben, direkt in den Prozess der Bildkompression zu integrieren.

3.1.2 State-of-the-Art

Bevor das in dieser Arbeit neu entwickelte Systemkonzept zur Überprüfung der Echtheit von Bildern erläutert wird, gibt dieser Abschnitt einen Überblick bekannter Verfahren zur Einbettung eines semi-fragilen, trägerbildabhängigen Wasserzeichens. Die Leistungsfähigkeit des neuen Systems wird in Abschnitt 3.3.5 mit den Daten dieser Verfahren verglichen.

JPEG-basierte Authentifizierung von Lin und Chang

Lange Zeit erschienen immer wiederkehrend Veröffentlichungen zu einem von den Autoren Lin und Chang vorgeschlagenen Wasserzeichenverfahren [LC98], [LC00] zur Authentifizierung von Bildern, basierend auf dem JPEG-Kompressionsstandard. Nach den für Bildkompression typischen Komponenten *Farbraumtransformation* und *Level-Offset* wird das Bild in 8x8 Pixel große Blöcke unterteilt und mithilfe der *Diskreten Cosinus Transformation* (DCT) in den Frequenzbereich überführt. Die entstehenden DCT-Koeffizienten werden entsprechend ihrer visuellen Relevanz quantisiert und anschließend entropiecodiert. Das Wasserzeichen zur Authentifizierung wird dabei während der Quantisierung eingebettet.

Die Autoren beschreiben eine invariante Beziehung zwischen den DCT-Koeffizienten derselben Position (u,v) in zwei getrennten 8x8-DCT-Blöcken $F_p(u,v)$ und $F_q(u,v)$ des transformierten Bildes, wobei $u,v \in \{0,...,7\}$. Diese Beziehung bleibt angeblich erhalten, wenn die Koeffizienten während einer erneuten JPEG-Kompression quantisiert werden. Der Algorithmus bildet aus den Größenverhältnissen einer Auswahl von Koeffizienten der Blöcke F_p und F_q eine Signatur, die anschließend durch eine Veränderung der Koeffizienten eingebettet wird.

Im Rahmen dieser Arbeit wurde das Verfahren von Lin und Chang implementiert und ausführlich analysiert. Dabei wurde festgestellt, dass die Größenbeziehungen der Blockkoeffizienten, anders als beschrieben, starken Veränderungen unterworfen sind (vgl. [SPPM05b]). Für die zur Authentifizierung verwendeten Koeffizienten sind demnach sehr große Toleranzbereiche notwendig, wodurch die Sicherheit des Systems deutlich reduziert wird.

In [MSCS06] greifen Maeno *et al*. das Verfahren von Lin und Chang erneut auf. Auch sie beschreiben das Sicherheitsproblem, das sich durch die großen Toleranzbereiche ergibt.

Dadurch bestätigen sie die im Rahmen der vorliegenden Arbeit bereits ein Jahr zuvor veröffentlichten Ideen [SPPM05b] zur Umgehung des in [LC00] vorgestellten Systems.

Korrelationsbasierte Authentifizierung von Fridrich

Ein anderes Verfahren zur Überprüfung der Echtheit von Bildern wurde in [Fri99] beschrieben. Es unterteilt das Trägerbild für die Generierung und Einbettung des Wasserzeichens in 64x64 Pixel-Blöcke. Jeder der Blöcke wird auf einen von 2^{30} Basisvektoren abgebildet. Die 30 Bit langen Basisvektoren dienen wiederum als Ausgangspunkt für die Erzeugung von 30 64x64 Pixel großen Rauschmustern, die mittels *Spread Spectrum Watermarking* in den DCT-Koeffizienten der 64x64 Pixel-Blöcke eingebettet werden.

Zur Authentifizierung bildet der Algorithmus erneut trägerbildabhängige, 64x64 Pixel große Rauschmuster. Werden diese per *Korrelation* (abzüglich Fehlertoleranz) in den DCT-Koeffizienten nachgewiesen, ist das Bild echt.

JPEG2000-basierte Authentifizierung in JPSEC

Das in JPSEC (vgl. Abschnitt 2.2.6) zum Einsatz kommende Authentifizierungssystem gliedert sich in drei Teile: eine verlustfreie, fragile und eine einerseits verlustfreie sowie andererseits verlustbehaftete, semi-fragile Authentifizierung (siehe Abbildung 3.2). Bei der fragilen Methode wird der JPEG2000-Bitstrom mithilfe eines traditionellen kryptografischen Signaturverfahrens geschützt. Jegliche Formatänderung, Transcodierung oder Bildverarbeitung führt zum Verlust der Integrität. Beim semi-fragilen Ansatz hingegen wird die Signatur nicht über alle Daten des Bitstromes gebildet. Es werden nur diejenigen Bits verwendet, die entstehen, wenn das Bild mit einer als *Lowest Authentication Bit Rate* (LABR) bezeichneten

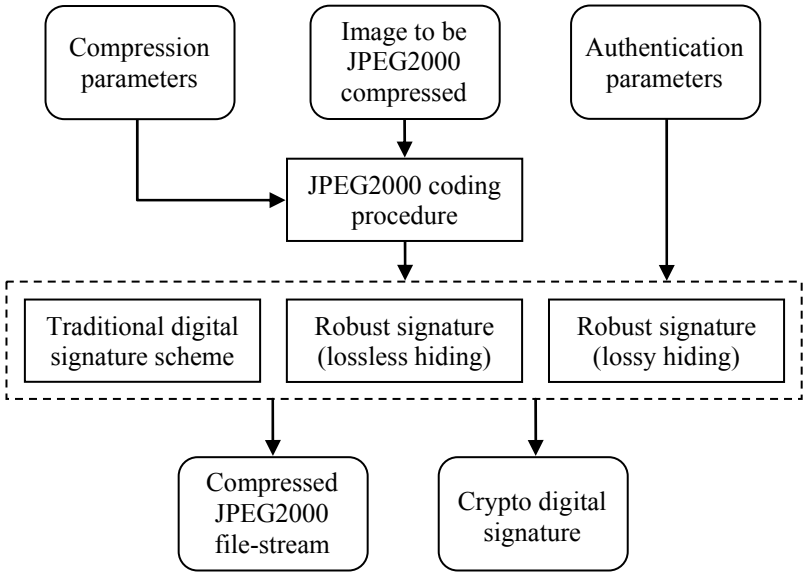


Abbildung 3.2 Systemstruktur der "Unified Authentication" im JPSEC-Standard (Quelle [ZQS+04])

Gesamtdatenrate codiert wird. Die Bits werden mittels Fehlerkorrektur gegen leichte Veränderungen geschützt. Die entstehenden Prüfbits der Fehlerkorrektur werden als Wasserzeichen im JPEG2000-codierten Bild eingebettet. Gleichzeitig werden alle erfassten Bits mithilfe einer Hash-Funktion und eines privaten Schlüssels zu einer Signatur verschlüsselt. Diese wird in allen drei Fällen parallel zum Bild oder in den Metadaten übermittelt bzw. gespeichert.

Der Unterschied zwischen der verlustfreien und der verlustbehafteten semi-fragilen Variante ist, dass erstere die FEC-Prüfbits (das Wasserzeichen) im Codestrom nach dem Verfahren von Ni *et al*. [NSA+04] einbettet, wohingegen die letztgenannte Variante das Verfahren von Sun und Chang [SC05] nutzt. Anders als die fragile JPEG2000-Authentifizierung sind diese beiden semi-fragilen Verfahren in der Lage, mithilfe der eingebetteten Prüfbits zu bestimmen, welche Codeblöcke des Bildes zu stark verändert bzw. manipuliert wurden. Dazu ermittelt die Verifizierungsoperation erneut einen Satz von Feature-Bits, der der vereinbarten LABR entspricht. Passen die Bits der Codeblöcke nach Anwendung der Fehlerkorrektur jeweils zu den extrahierten FEC-Prüfbits, so werden die Blöcke als korrekt angenommen, die Feature-Bits einer Hash-Funktion zugeführt und mit der entschlüsselten Signatur verglichen.

Die verlustfreie Wasserzeicheneinbettung basiert auf dem Patchwork-Ansatz von Bender [BGML96]. Die Menge der Einbettungspositionen des Trägers, bei JPSEC die DWT-Koeffizienten, wird pseudozufällig in zwei Gruppen G_A und G_B geteilt. Ist die Anzahl n der Gruppenelemente $g_A \in G_A$ bzw. $g_B \in G_B$ hinreichend groß, so ist statistisch gesehen zu erwarten, dass deren Mittelwerte gleich sind, d.h. $\bar{g}_A - \bar{g}_B = 0$. Ebenso ist zu erwarten, dass die Verteilung der Werte um den Mittelwert in beiden Gruppen gleich ist. Das Bit-Einbetten erfolgt, indem diese statistische Verteilung der Werte geändert wird. Beim Extrahieren des Wasserzeichens kann diese "statistisch untypische" Veränderung erkannt werden. Durch das Zurücksetzen der Veränderungen auf Seiten der Extraktion ist das System reversibel (verlustfrei). Eine nähere Erläuterung findet sich auch in [KXYM07].

Die verlustbehaftete Wasserzeicheneinbettung in JPSEC hingegen verändert die DWT-Koeffizienten irreversibel. Während des EBCOT-Prozesses, also vor der JPEG2000-Codestromgenerierung, werden einzelne Codebits durch die FEC-Prüfbits ersetzt[14].

Vom Bildinhalt unabhängige Authentifizierung

Weiterhin existieren Verfahren zur Überprüfung der Echtheit von Bildern, die ein unabhängig vom Bildinhalt generiertes Wasserzeichen im Träger einbetten. Beispiele[15] finden sich in

[14] Für die JPSEC-Authentifizierung existieren keine Angaben zur Leistungsfähigkeit. Eine eigene Implementierung konnte im Rahmen dieser Arbeit nicht vorgenommen werden. Ein konkreter Leistungsvergleich entfällt.

[15] Diese Verfahren werden aufgeführt, da sie in der Literatur für die Authentifizierung von Bildern wiederkehrend zitiert und für Leistungsvergleiche benutzt werden.

[LPD00], [EG01], [KH99], [Que01] und [YLL01]. Um zu verhindern, dass ein Angreifer das Trägerbild unabhängig vom Wasserzeichen beliebig manipulieren kann, werden die Einbettungspositionen oft mithilfe eines geheimen Schlüssels geschützt. Um die Echtheit überprüfen zu können, muss der Schlüssel jedoch bekannt sein. An dieser Stelle ergibt sich eine Sicherheitslücke. Die in Abschnitt 3.1.1 definierten Anforderungen "Abhängigkeit und Öffentliche Verifizierung" können durch diese Verfahren nicht erfüllt werden.

3.2 Systemkonzept der semi-fragilen Bildauthentifizierung

In diesem Abschnitt wird das im Rahmen dieser Arbeit entwickelte Wasserzeichenverfahren zur Authentifizierung von Bildern erläutert. Die einzelnen Komponenten *Feature-Extraktion*, *Wasserzeicheneinbettung* und -*extraktion* sowie *Bildrekonstruktion* und *Fehlerkorrektur* werden detailliert beschrieben. Im Anschluss daran erfolgt im Abschnitt 3.3 eine ausführliche Analyse der Manipulationssicherheit und Robustheit gegenüber erlaubten Operationen sowie ein Vergleich der Leistungsfähigkeit mit Verfahren anderer Autoren.

Das Gesamtsystem der entwickelten Authentifizierung ist in Abbildung 3.3 dargestellt. Zuerst wird mithilfe einer Hash-Funktion der Bildinhalt sicher erfasst. Dazu werden jedoch nicht die einzelnen Grauwerte des Bildes direkt genutzt, sondern aus dem Bild ermittelte semi-fragile Eigenschaften (engl. *features*). Anschließend wird der Hash-Wert[16] verschlüsselt (signiert), mit einem Fehlerschutz versehen und als Wasserzeichen eingebettet. Auf Seiten der Extraktion wird das Wasserzeichen ausgelesen, entschlüsselt und mit dem erneut aus den Bildeigenschaften gebildeten Hash verglichen. Stimmen beide Werte überein, ist das Bild echt.

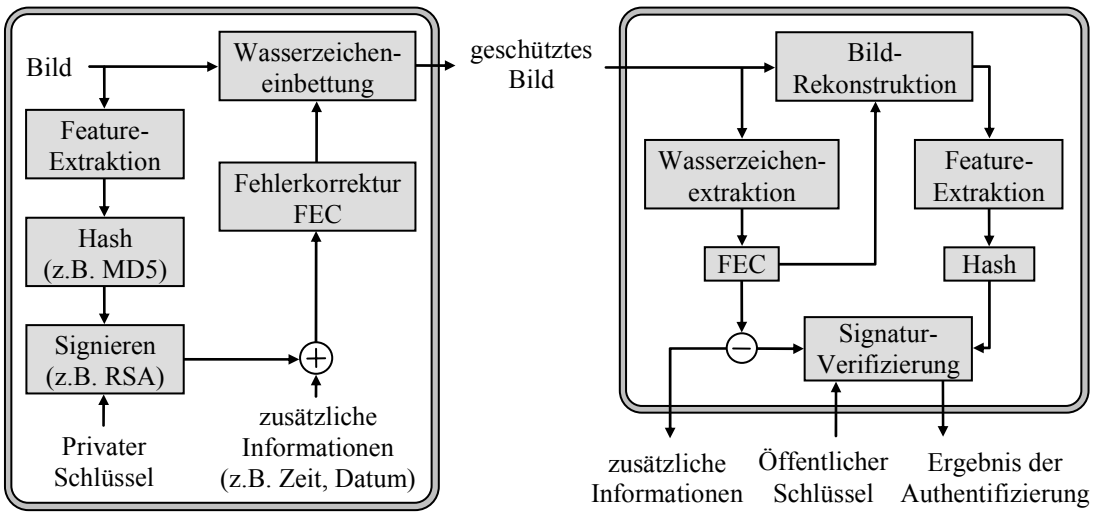


Abbildung 3.3 Digitales Wasserzeichensystem zur Überprüfung der Echtheit von Bildern

[16] Sequenz mit einer Länge von nur einigen hundert Bits

3.2.1 Generierung und Einbettung des JPEG2000-basierten Wasserzeichens

Das entwickelte System basiert auf der Quantisierung der DWT-Koeffizienten[17] des Trägerbildes und ist in den Kompressionsprozess einer JPEG2000-Codierung (siehe Abschnitt 2.2) integriert. Die Auswahl der DWT-Koeffizienten als Einbettungsdomain, der Quantisierung als Einbettungstechnik und der JPEG2000-Kompression als zugrunde liegendes *Container-Format* wurde aus folgenden Gründen getroffen:

- JPEG2000 ist ein aktuelles, leistungsstarkes Kompressionsverfahren mit einer Vielzahl von Skalierungseigenschaften (Zugriff auf Teilbildbereiche, Auflösungs- und Qualitätsstufen, ohne den gesamten Bitstrom decodieren zu müssen). Der klassische JPEG-Standard bietet diese Eigenschaften nicht. Alternativ das Wasserzeichen vor der Kompression zu generieren und einzubetten, ist nicht besonders effizient. Darüber hinaus können das Wasserzeichen störende Einflüsse der Bildkompression besser behandelt werden, wenn Feature-Extraktion und Einbettung in den Kompressionsprozess integriert sind.
- Die DWT ist Bestandteil der JPEG2000-Kompression.
- Beim Direct Embedding (siehe Abschnitt 2.1.1) handelt es sich um eine auf Quantisierung basierende Einbettungstechnik für Digitale Wasserzeichen. Da die Quantisierung der DWT-Koeffizienten ein integraler Bestandteil der JPEG2000-Kompression ist, kann sie gleichfalls effizient zum Einbetten genutzt werden. Zudem wird die Quantisierung, wie im Folgenden beschrieben, eingesetzt, um Bildveränderungen in erlaubte und nicht erlaubte Operationen zu unterteilen (Anforderung: Semi-Fragilität).

Konstruktion eines sicheren bildabhängigen Hash-Wertes

Die den Bildinhalt beschreibenden Features sollen manipulationssicher sein und trotzdem erlaubte Bildveränderungen zulassen. Eine einfache Möglichkeit, diese beiden Eigenschaften zu erreichen, ist die Quantisierung der Werte $x := \{x_n \in \mathbb{R} : 1 \le n \le N\}$, die das zu schützende Bild in einer beliebigen Domain repräsentieren. Dazu wird ein Wert x, wie in Abschnitt 2.2 beschrieben, durch die Quantisierung $Q(\bullet)$ mit der Schrittweite Δ zum Wert q quantisiert (vgl. Abbildung 2.9). Wenn eine Störung den rekonstruierten Wert $\hat{x} = Q^{-1}(q)$ nicht stärker als $\Delta/2$ verändert[18], wird er bei einer erneuten Quantisierung mit derselben Schrittweite ebenfalls zum Wert q quantisiert. Werden alle Werte des Bildes mit der Schrittweite Δ quantisiert und einer Hash-Funktion zugeführt, sind somit alle Bildoperationen zugelassen, die den Betrag der Werte $\hat{x}$ nicht über das jeweilige Intervall $\lfloor \Delta\mathbb{Z}; \Delta(\mathbb{Z}+1) \rfloor$ hinaus verändern.

[17] Koeffizienten der *Diskreten Wavelet-Transformation* (DWT)

[18] Der Bereich $[-\Delta; +\Delta]$ bildet eine Ausnahme. Er ist doppelt so groß wie die übrigen Quantisierungsintervalle, da es sich um eine Quantisierung mit Totzone handelt.

In dieser Arbeit werden die DWT-Koeffizienten des Trägerbildes zur Konstruktion eines bildabhängigen Hash-Wertes genutzt. Mithilfe von Simulationen wurde untersucht, wie stark sich die Koeffizienten unterschiedlicher DWT-Zerlegungsstufen aufgrund von Störungen verändern (siehe Anhang B.1.1). Die Simulationsergebnisse zeigen, dass die HL-, LH- und HH-Koeffizienten eines DWT-Subbandes im Gegensatz zu den LL-Koeffizienten auch durch eine Helligkeits- oder Kontraständerung des Trägerbildes nur wenig beeinflusst werden. Beispielsweise können die HL_4-, LH_4- und HH_4-Koeffizienten der Zerlegungsstufe $Z = 4$ mit einer Schrittweite $\Delta = 4$ quantisiert werden, um gegen erlaubte Störungen[19] "weitestgehend" robust zu sein. Die exakte Robustheit wurde durch Simulationen ermittelt, deren Ergebnisse sich im Anhang B.3 befinden. Abbildung 3.4 zeigt z.B. die Fehlerwahrscheinlichkeiten dafür, dass ein quantisierter HL_4-, LH_4- und HH_4-Koeffizient nach der Anwendung von Störungen nicht wieder zum selben Intervall quantisiert wird. Derartige Fehler führen beim Einsatz einer Hash-Funktion zur Veränderung des gesamten Hash-Wertes für das entsprechende DWT-Subband. Als Konsequenz dessen wird das Bild als unecht ausgegeben.

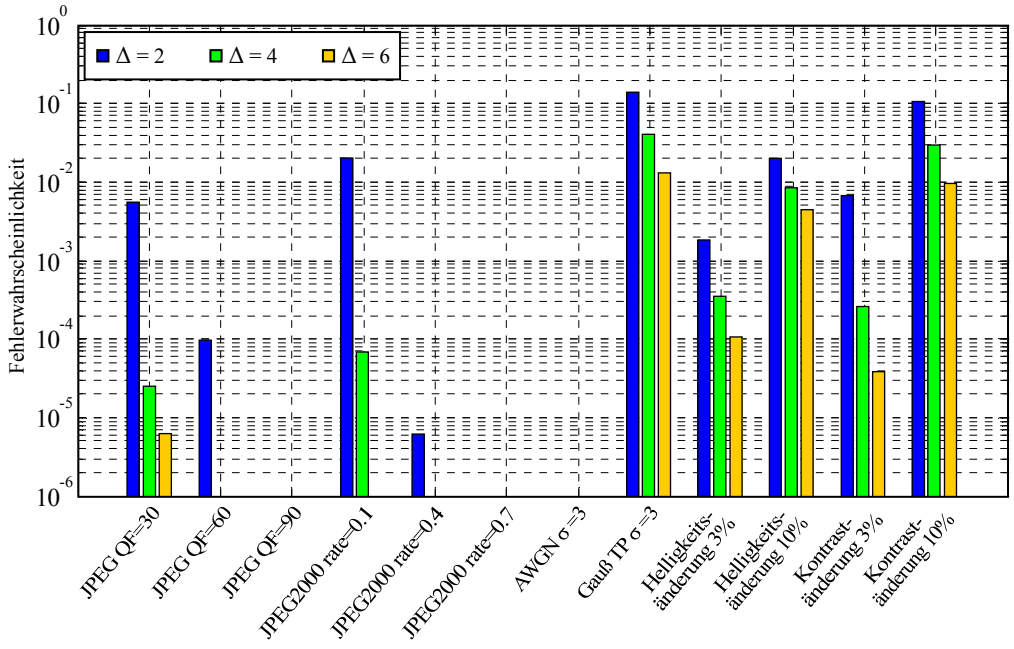


Abbildung 3.4 Auszug der Simulationsergebnisse (Anhang B.3): Wahrscheinlichkeit für die Quantisierung eines der HL_4-, LH_4-, HH_4-Koeffizienten zu einem falschen Quantisierungsinvertall infolge von Störungen

Modifizierte Quantisierung im Bereich der Totzone

Je größer die eingesetzte Quantisierungsschrittweite Δ ist, desto stärker sind die Verzerrungen des Trägerbildes, die aus der Quantisierung der DWT-Koeffizienten resultieren. Bei einer

[19] Als Störungen werden hier Bildoperationen bezeichnet, gegen die die Authentifizierung robust sein soll. Dazu zählen in den Simulationen dieser Arbeit die verlustbehaftete Kompression, additives Rauschen, Tiefpassfilterung, Helligkeits- und Kontraständerungen, Bildschärfung sowie eine Skalierung der Größe des Trägerbildes.

Quantisierung der LL_4-Koeffizienten des vierten DWT-Subbandes mit der Schrittweite $\Delta = 4$ ergibt sich bspw. im Mittel für die verwendeten Testbilder ein PSNR von 46,57 dB[20]. Jedoch führt die in Abschnitt 2.2 beschriebene Quantisierung bei den HL-, LH- und HH-Koeffizienten zu starken Verzerrungen (z.B. HL_4-, LH_4- und HH_4-Subband quantisiert mit $\Delta = 4$ ergibt ein PSNR von 30,68 dB). Diese Koeffizienten besitzen im Gegensatz zu den LL-Koeffizienten meist kleine Werte. Werden viele davon zu Null gesetzt, gehen wichtige Bildinformationen verloren. Aus diesem Grund wurde in dieser Arbeit eine neue Quantisierung zur Hash-Wert-Berechnung entwickelt[21], die alle DWT-Koeffizienten, deren Werte im Bereich $[-\Delta/2;+\Delta/2]$ der Totzone liegen, unverändert lässt. Der bei $\Delta = 4$ aus der Quantisierung der HL_4-, LH_4- und HH_4-Koeffizienten resultierende, gemittelte PSNR-Wert kann dadurch von 30,68 dB auf 44,97 dB gesteigert werden.

Empirische Untersuchungen und Simulationen zur Manipulationssicherheit haben gezeigt, dass es genügt, die Authentifizierungssignatur ausschließlich anhand der quantisierten LL_4-Koeffizienten der vierten DWT-Zerlegungsstufe zu bilden. Auf diese Weise konnte ein guter Kompromiss zwischen Robustheit, Trägerbildqualität und Sicherheit gefunden werden (siehe Abschnitt 3.3).

Einbettung des Wasserzeichens durch Quantisierung der DWT-Koeffizienten

Bei semi-fragilen Wasserzeichenverfahren zur Authentifizierung muss die eingebettete, den Bildinhalt beschreibende Signatur theoretisch nur so robust sein wie die Features, aus denen die Signatur gebildet wird. Das heißt, wenn die Features durch einen Angriff so weit verändert werden, dass der aus ihnen gebildete Hash-Wert verändert wird, ist es uninteressant, ob die eingebettete Signatur noch korrekt ist. Die Verifizierung schlägt ohnehin fehl und das Bild wird als unecht gemeldet.

Wird die Signatur nur so stark im Träger eingebettet wie nötig, kann die durch das Einbetten auftretende Bildverzerrung gering gehalten werden. Es ist also sinnvoll, das Wasserzeichen direkt in den Features einzubetten, aus denen auch die Signatur berechnet wird. Wie bereits von Fei *et al*. in [FKK06] festgestellt wurde, ist es darüber hinaus wichtig, dass der Bereich, in dem die Authentifizierungsdaten eingebettet werden, nicht ungeschützt gelassen werden darf. Das heißt, die Trägerpositionen, an denen das Wasserzeichen eingebettet wird, müssen unbedingt auch von der Hash-Berechnung eingeschlossen werden. Andernfalls wäre es einem Angreifer möglich, die durch diese Positionen repräsentierte Bildinformation zu verändern, solange das Wasserzeichen gleich bleibt.

[20] Die ausführlichen Simulationsergebnisse befinden sich im Anhang B.2.

[21] vgl. [SPM06]

Zum Einbetten des Wasserzeichens wird in dieser Arbeit die in Abschnitt 2.1.1 beschriebene *Dither Modulation* verwendet. Jedoch wurde sie so verändert, dass sie der in Abschnitt 2.2 beschriebenen Quantisierung mit Totzone entspricht (vgl. [SPM06]). Das Einbetten mithilfe der veränderten eindimensionalen *Dither Modulation* mit Totzone folgt nun Gleichung (3.1).

$$y(p_k) = \begin{cases} x(p_k) & -\Delta/4 \le x(p_k) \le \Delta/4 \\ \Delta \cdot \left(\operatorname{sign}(x(p_k)) \left\lfloor \frac{|x(p_k)|}{\Delta} \right\rfloor + w_k \cdot 1/4 + d_m \right) & \text{sonst} \end{cases} \tag{3.1}$$

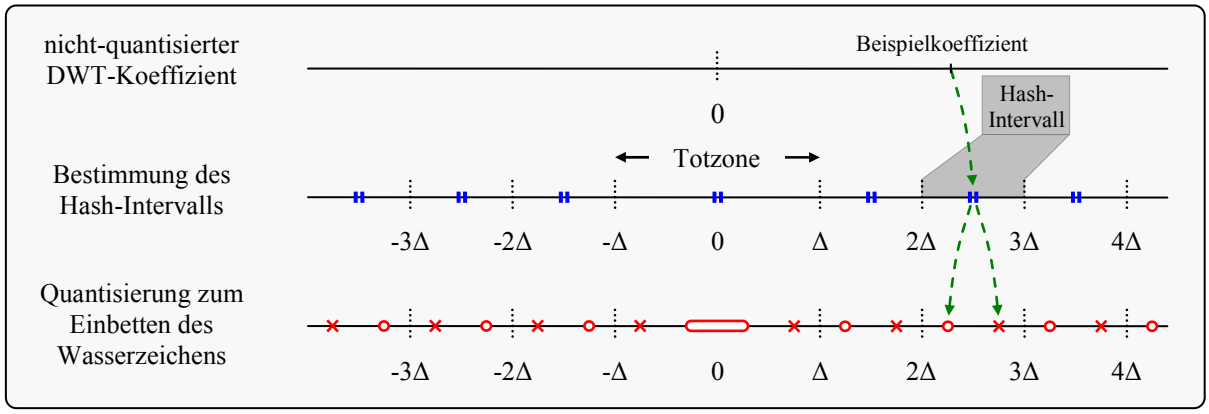


Abbildung 3.5 Beispiel für die Quantisierung zur Bestimmung des Hash-Intervalls und zum Einbetten mithilfe der veränderten Dither Modulation mit Totzone

Die Trägersignalwerte werden zu den dichtesten Quantisierungsgitterpunkten jeweils eines der zwei Teilgitter $\Lambda_{w_k} := \{\Delta\mathbb{Z} + w_k\Delta/4\}$ (Punkte × bzw. o in Abbildung 3.5) hin verändert, wobei $w_k = \{-1, +1\}$ das an der Stelle p_k einzubettende Wasserzeichenbit darstellt. Zur Einbettung wird in dieser Arbeit das gröbste DWT-Subband (die LL-Band-Koeffizienten[22]) ausgewählt. Die Verwendung dieses Subbandes führt im Gegensatz zu den HL-, LH- und HH-Koeffizienten zu geringeren Verzerrungen des Trägerbildes[23]. Allerdings bietet das Einbetten in den LL-Band-Koeffizienten keine Robustheit gegenüber Helligkeits- und Kontraständerungen, woraufhin das Verfahren um folgende Normierung erweitert wurde.

Helligkeits- und Kontrastnormierung des Trägerbildes

Um zusätzlich Robustheit gegenüber Helligkeits- und Kontraständerungen zu erhalten, wurde in dieser Arbeit eine Bildnormierung entwickelt, die während der Quantisierung der DWT-Koeffizienten stattfindet. In einem ersten Schritt werden die Koeffizienten des LL-Bandes auf

[22] Für die verwendeten Testbilder (512 x 512 Pixel) dienen die Koeffizienten des LL_4-Bandes als Trägersignal.

[23] vgl. Bildverzerrungen durch die Quantisierung der DWT-Koeffizienten im Anhang B.2

die mittlere Bildhelligkeit normiert (Subtraktion des *einfachen Mittelwertes*[24]). In einem zweiten Schritt wird die Quantisierungsschrittweite zum Generieren und Einbetten des Wasserzeichens auf den Bildkontrast normiert. Dazu wird ein Normierungsfaktor g berechnet (siehe Gleichung (3.2)), durch den die Schrittweite Δ dividiert wird. Als Ideengrundlage dienten die Untersuchungen zur *Rational Dither Modulation* von Pérez-González *et al*. in [PMBA04].

$$g = \frac{1}{256}\left(\frac{1}{N}\sum_{n=1}^{N} I_n^{\,2}\right)^{1/2} \tag{3.2}$$

Nach der Quantisierung der DWT-Koeffizienten zum Generieren und Einbetten des Wasserzeichens werden diese zurücknormiert. Die Koeffizienten des LL-Bandes werden mit dem Normierungsfaktor g multipliziert. Die Addition des zuvor berechneten Mittelwertes führt dann wieder auf den ursprünglichen Wertebereich. Eine Kontraständerung des Bildes kann anschließend als eine direkte Skalierung des gebildeten Normierungsfaktors g angesehen werden. Durch die Division der Koeffizienten des gestörten Trägerbildes $\tilde{I}$ während der Wasserzeichenextraktion durch den erneut gebildeten Normierungsfaktor $\tilde{g}$ wird die Kontraständerung aufgehoben. Helligkeitsänderungen wiederum werden durch die erneute Subtraktion des mittleren Grauwertes behandelt.

Simulationen haben gezeigt, dass diese Normierung eine deutliche Erhöhung der Robustheit der DWT-Koeffizienten des LL-Bandes sowohl gegenüber Kontrast- als auch Helligkeitsänderungen ermöglicht (siehe Anhang B.1.2). Jedoch wird die Robustheit gegenüber Gaußscher Tiefpassfilterung und Bildschärfungsoperationen durch diese Normierung geschwächt. Diese beiden Operationen "widersprechen" der Kontrast- und Helligkeitsnormierung. Um an dieser Stelle keinen Kompromiss eingehen zu müssen, wird neben der Einbettung des Authentifizierungswasserzeichens eine zweite Einbettung vorgenommen (siehe Abbildung 3.6).

Der Normierungsfaktor g wird mittels Fehlerkorrektur (*Repeat-Accumulate Coding*) codiert[25] und mit geringer Quantisierungsstärke[26] in den HL_4-, LH_4- und HH_4-Koeffizienten der DWT-Zerlegungsstufe $Z = 4$ eingebettet. Hierdurch entstehen keine wahrnehmbaren Verzerrungen.

Bei der Echtheitsüberprüfung des Bildes wird das eingebettete Authentifizierungswasserzeichen einmal für die, aus dem empfangenen Bild $\tilde{I}$ erneut berechnete Kontrastnormierung $\tilde{g}$ und ein zweites Mal für die extrahierte Kontrastnormierung $\hat{g}$ verifiziert. Ist das Ergebnis

---

[24] einfacher Mittelwert $\frac{1}{N}\sum_{n=1}^{N} x_n$ der DWT-Koeffizienten x des LL-Bandes (Trägersignal)

[25] Codierung des Normierungsfaktors g (Gleitkommadarstellung: 32 Bit) erfolgt mit einer Coderate $r = 1/96$

[26] entspricht $1/5$ der Stärke für die Einbettung des Authentifizierungswasserzeichens in den LL_4-Koeffizienten (empirisch ermittelt hinsichtlich Bildverzerrungen nach Quantisierung der HL_4-, LH_4- und HH_4-Koefiizienten)

mindestens einer der beiden Verifizierungsoperationen positiv, dann wird das Bild als echt ausgegeben. Dabei führt die erneut berechnete Normierung $\tilde{g}$ zu einer Steigerung der Robustheit gegenüber Helligkeits- und Kontraständerungen[27], während die extrahierte Normierung $\hat{g}$ robust gegenüber Tiefpassfilterung und Bildschärfung ist.

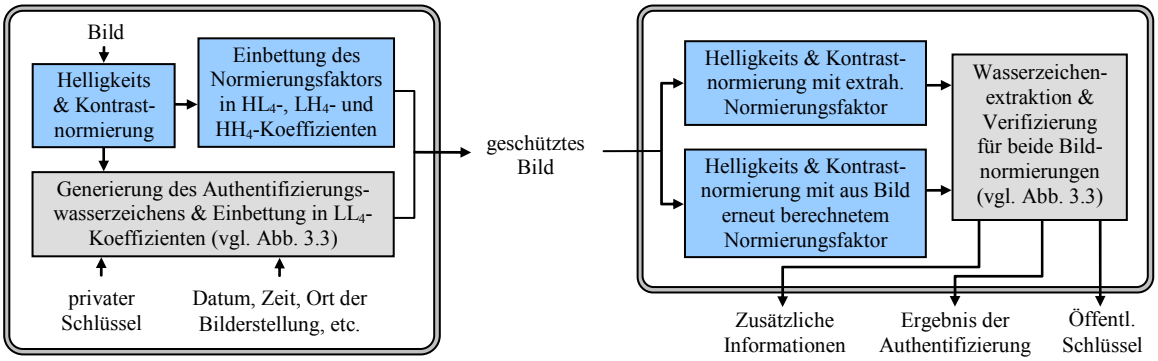


Abbildung 3.6 Entwickeltes digitales Wasserzeichensystem zur Echtheitsüberprüfung von Bildern erweitert um Kontrast- und Helligkeitsnormierung

3.2.2 Extraktion des Wasserzeichens und Rekonstruktion des Trägerbildes

Zur Überprüfung der Echtheit eines geschützten Bildes muss das Wasserzeichen aus den DWT-Koeffizienten des LL-Bandes extrahiert werden. Die darin enthaltene Signatur muss zusammen mit dem erneut aus den Koeffizienten gebildeten Hash-Wert verifiziert werden. Stimmt der aus der Signatur mithilfe des öffentlichen Schlüssels entschlüsselte Hash-Wert mit dem erneut gebildeten Wert überein, dann wird das Bild als echt gemeldet.

Die Wasserzeichenextraktion geschieht, wie in Abschnitt 2.1.1 beschrieben, durch eine Zuordnung der Trägersignalwerte zum jeweils nächstgelegenen Gitterpunkt × bzw. o des Gesamtquantisierungsgitters $\Lambda = \Lambda_{-1} \cup \Lambda_{+1}$. Die Wasserzeichenbits werden dabei durch Gleichung (2.3) aus dem Träger (DWT-Koeffizienten des LL-Bandes) ermittelt.

Verbesserung der Bildqualität durch Rekonstruktion der Hash-Intervalle

Wie in Abbildung 3.5 dargestellt, führt das Einbetten der Wasserzeichenbits zu Verzerrungen der DWT-Koeffizienten in die Mitte der unteren bzw. oberen Hälfte des jeweiligen Quantisierungsintervalls. Das heißt, die Koeffizienten werden auf die Werte $\Delta(\mathbb{Z}+1/4)$ bzw. $\Delta(\mathbb{Z}+3/4)$ gesetzt. Nach der Extraktion des Wasserzeichens können diese Koeffizienten von der unteren bzw. oberen Hälfte wieder in die Mitte $\Delta(\mathbb{Z}+1/2)$ des Intervalls gesetzt werden. Auf diese Weise wird das Wasserzeichen aus dem Träger entfernt und die durch das Einbetten verur-

[27] Wenn das geschützte Bild durch *Clipping* über den Grauwertebereich $[0;255]$ hinaus manipuliert wird, schlägt die Verifizierung fehl. Die verursachten Veränderungen der DWT-Koeffizienten des LL-Subbandes sind zu groß, wie Abbildung B.29 in Anhang B.1.2 beispielhaft beweist.

sachten Bildverzerrungen verringert. Anhand von Simulationen (siehe Anhang C.1) wurde ermittelt, dass eine derartige Bildrekonstruktion unabhängig von der Quantisierungsschrittweite zu einer Verringerung der Bildverzerrungen (Erhöhung des PSNR um ca. 2,5 dB) führt.

Entwickelte Strategie zur Skalierungsrobustheit

Steht durch eine JPEG2000-Transcodierung oder während einer auflösungsprogressiven Decodierung nur ein Teil des Bitstroms zur Verfügung (siehe Abbildung 3.7 links), kann die Echtheit für diesen Teil trotzdem bestimmt werden. Wird das komprimierte Bild jedoch zwischendurch dekodiert, skaliert und evtl. in ein anderes Kompressionsformat gewandelt, geht die Information über die DWT-Zerlegungsstufen aufgrund unbekannter Originalauflösung verloren. Um dennoch robust gegenüber Änderungen der Bildauflösung zu sein, wurde die in dieser Arbeit entwickelte Authentifizierung wie folgt erweitert. Vor der DWT-Zerlegung des Trägerbildes wird dieses sowohl beim Generieren und Einbetten als auch beim Extrahieren und Verifizieren der Wasserzeichensignatur auf eine feste Größe von 512x512 Pixel normiert. Nach dem Einbetten wird das Trägerbild wieder auf die ursprüngliche Größe skaliert.

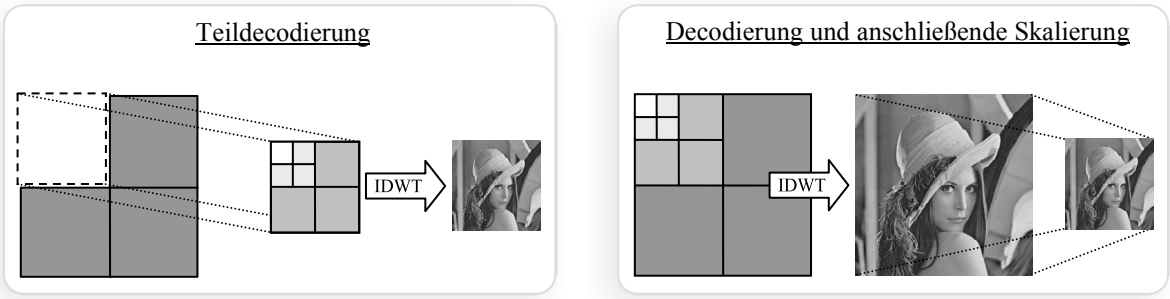


Abbildung 3.7 Decodierung des Bildes in skalierter Auflösung (links) bzw. Decodierung mit anschließender Skalierung (rechts)

Eine Veränderung der Größe des geschützten Bildes wirkt sich durch die Größennormierung ausschließlich in Form von Interpolationsfehlern auf die Koeffizienten der vierten DWT-Zerlegungsstufe aus. Die auf diese Weise erzielte Robustheit gegenüber Bildskalierungen wird anhand der Simulationsergebnisse in Anhang C.2 demonstriert.

3.2.3 Fehlerkorrektur des Hash-Intervalls zur Erhöhung der Robustheit

Mit dem Einsatz der Dither Modulation zum Einbetten des Wasserzeichens an den vorgesehenen Trägersignalpositionen sinkt die Robustheit des Hash-Wertes, der durch Quantisierung der Koeffizienten ebenfalls an diesen Positionen zuvor gebildet wurde. Wie in Abschnitt 3.2.1 beschrieben, ist der Hash-Wert robust, solange die Beträge der quantisierten DWT-Koeffizienten durch Störungen nicht über das jeweilige Intervall $\left[\Delta\mathbb{Z};\Delta\left(\mathbb{Z}+1\right)\right)$ hinaus verändert werden. Durch das Einbetten werden die Koeffizienten jedoch von der Mitte (Punkt ▮▮ in

Abbildung 3.5) in die untere bzw. obere Hälfte des Hash-Intervalls gesetzt. Infolgedessen reicht eine Störung um den Betrag $\Delta/4$ aus, um den Hash-Wert zu verändern.

Um die Robustheit zu steigern, wurde in dieser Arbeit eine Strategie entwickelt [SPPM05a], die die Fehlerkorrektur der Wasserzeichenbits mit der Bildung des Hash-Wertes verbindet. Angenommen, vor dem Einbetten des Wasserzeichens w wurde dieses mithilfe einer Fehlerkorrektur[28] codiert. Sei $\hat{w}_k = \{-1,+1\}$ ein aus dem gestörten Trägersignal $\hat{y} \in \mathbb{R}$ an der Position p_k extrahiertes und $\tilde{w}_k = \{-1,+1\}$ das nach Anwendung der FEC-Decodierung korrigierte Wasserzeichenbit. Um mit dieser Fehlerkorrektur ebenfalls Störungen des Hash-Intervalls korrigieren zu können, wird Gleichung (3.3) genutzt.

$$q_k = \operatorname{sign}(\hat{y}_k) \cdot \left\lfloor \frac{|\hat{y}_k|}{\Delta} + \frac{\hat{w}_k - \tilde{w}_k}{8} \right\rfloor \tag{3.3}$$

Wie in Abbildung 3.8 dargestellt, wird das rekonstruierbare Hash-Intervall durch die Fehlerkorrektur, abhängig vom Wasserzeichenbit w_k, auf den Bereich $\left[\Delta(\mathbb{Z}-1/4);\Delta(\mathbb{Z}+3/4)\right)$ bzw. $\left[\Delta(\mathbb{Z}+1/4);\Delta(\mathbb{Z}+5/4)\right)$ ausgedehnt. Dadurch wird der Hash-Wert erst von Störungen beeinflusst, deren Betrag größer als $\Delta/2$ ist. Die Robustheit wurde auf diese Weise verdoppelt, was anhand von Simulationen bestätigt werden konnte (siehe Anhang C).

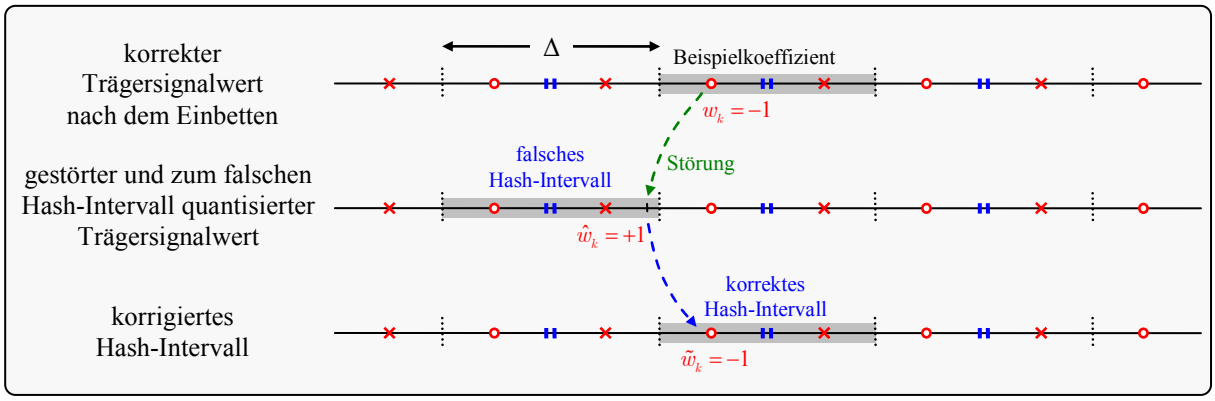


Abbildung 3.8 Beispiel für die Rekonstruktion des Hash-Intervalls durch Ausweitung der Bitfehlerkorrektur auf die Quantisierung zur Bildung des Hash-Wertes

3.3 Leistungsanalyse des entwickelten Authentifizierungssystems

Nach einer Erläuterung der Simulationsergebnisse hinsichtlich Robustheit gegenüber erlaubten Störungen (Abschnitt 3.3.2) erfolgt in Abschnitt 3.3.3 eine Erörterung der Manipulationssicherheit der entwickelten Authentifizierung gegenüber nicht erlaubten Störungen. Im An-

[28] Zur Fehlerkorrektur wird die in Abschnitt 2.3 beschriebene Faltungscodierung mit einer Coderate R_c = 1/2 und ein Viterbi-Decoder mit *soft-decision input* eingesetzt.

schluss daran werden die Eigenschaften und Leistungsdaten mit Verfahren anderer Autoren verglichen. Dabei dienen in erster Linie die aufgearbeiteten Angaben aus [ESC+04] und die durch eigene Implementierungen und Simulationen ermittelten Robustheitsergebnisse.

3.3.1 Implementierung der Algorithmen

Neben der Implementierung der entwickelten Algorithmen zu Simulationszwecken in der Programmierumgebung *MATLAB* erfolgte im Rahmen dieser Arbeit eine vollständige Integration der DWT-basierten Authentifizierungslösung in einen JPEG2000-Kompressionsprozess. Dazu wurde ein bestehender Quellcode der in Abschnitt 2.2 beschriebenen JPEG2000-Komponenten verwendet [TM02]. Dieser Code ist in der Programmiersprache *C* verfasst und trägt den Namen *Kakaku*. Die Wahl fiel vor allem deshalb auf diese Software, weil sie auf Laufzeit und Ressourcenbedarf optimiert ist und sich gut für eine Umsetzung auf kleinen, mobilen Geräten wie bspw. Pocket-PC[29] eignet.

3.3.2 Simulationsergebnisse der Robustheitstests

Die Leistungsfähigkeit eines Systems zur Bildauthentifizierung wird vorrangig daran gemessen, bei erlaubten, den Bildinhalt nicht beeinflussenden Veränderungen keinen Fehlalarm auszulösen. Sowohl die Berechnung des Hash-Wertes als auch das eingebettete Wasserzeichen müssen folglich robust gegenüber einer Vielzahl unterschiedlicher Störungen sein, die im gewohnten Umgang mit Bildern zwangsläufig entstehen oder der Bildverbesserung dienen.

In dieser Arbeit wurde eine Auswahl der folgenden acht Bildoperationen getroffen, gegenüber derer die Robustheit umfangreich getestet wurde:

- verlustbehaftete JPEG-Kompression (Parameter: Quality Factor 0…100)
- verlustbehaftete JPEG2000-Kompression (Parameter: Target Rate 0…1 bpp)
- additives Gaußsches Rauschen (Parameter: Standardabweichung σ)
- Gaußsche Tiefpassfilterung (Filtergröße 3x3, Parameter: Standardabweichung σ)
- Veränderung der Bildhelligkeit (Parameter: Pixel-Grauwerte $-50\ldots+50$)
- Veränderung des Bildkontrastes (Parameter: Verstärkungsfaktor 0,5…1,5)
- Veränderung der Bildauflösung (Parameter: Skalierungsfaktor 0,25…2)
- Bildschärfung durch *Gauß-Unsharp-Masking*
  (Filtergröße 3x3, Parameter: Standardabweichung σ)

[29] Während des Verfassens dieser Arbeit ist im Landesforschungsschwerpunkt IuK (Verbundforschungsprojekt M6C - UR 02 025 / 10 / 2004) eine Pocket-PC-Implementierung der Authentifizierungslösung entstanden.

Abbildung 3.9 zeigt eine Zusammenfassung der Simulationsergebnisse aus Anhang C.2, wobei die Anzahl der fehlgeschlagenen Verifikationen im Verhältnis zur Anzahl der gesamt getesteten Bilder dargestellt ist (FRP = *false positive ratio*). Es ist deutlich zu erkennen, dass die Robustheit des entwickelten Verfahrens bei zunehmender Quantisierungsschrittweite Δ größer wird. Jedoch vergrößern sich im gleichen Zuge dabei auch die Verzerrungen des Trägerbildes. Bei einer Schrittweite $\Delta = 4$ ergibt sich bspw. im Mittel ein PSNR von 43,51 dB[30].

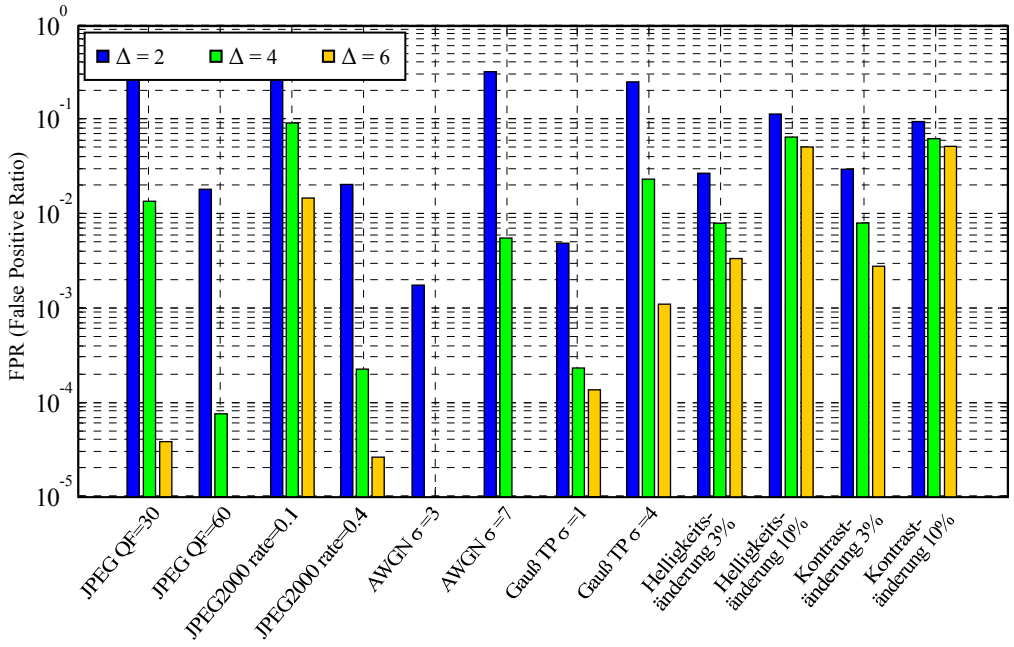


Abbildung 3.9 Robustheit des entwickelten Authentifizierungssverfahrens gegenüber JPEG-/JPEG2000-Kompression, Helligkeits-/Kontraständerungen sowie Gaußschem Rauschen und Tiefpassfilterung

Der Grund für die trotz Normierung teilweise geringe Robustheit einiger Bilder gegenüber Helligkeitsänderungen und Kontrasterhöhungen ist die Bereichsübersteuerung[31] der Grauwerte. Werden Grauwerte des Trägerbildes durch eine dieser beiden Operationen über den zulässigen Bereich $[0;255]$ (bei 8 Bit) hinaus verändert, gehen Bildinformationen verloren. Dadurch verändert sich der Hash-Wert des Bildes und die Verifikation schlägt fehl.

3.3.3 Analyse der Manipulationssicherheit

Neben der Robustheit des eingebetteten Authentifizierungswasserzeichens gegenüber erlaubten Störungen ist es ebenso wichtig, dass die Verifizierung Alarm auslöst, wenn nicht erlaubte Störungen (Attacken) auf das Bild angewendet wurden. In der Literatur werden Bilder, die nach der Anwendung erlaubter Störungen fälschlicherweise als unecht erkannt werden, als

[30] Ausführliche Simulationsergebnisse sind Abbildung C.1 (Anhang) zu entnehmen.

[31] engl. *Clipping*

false positives bezeichnet. Hingegen inhaltlich manipulierte Bilder, die die Verifizierung ohne Alarm passieren, heißen *false negatives*.

Oft ist jedoch eine Trennung erlaubter und nicht erlaubter Bildverarbeitung schwierig. Wie in Abbildung 3.10 dargestellt, existiert ein unscharfer Bereich zwischen authentischen und nicht-authentischen Bildern, dessen Grenzen nicht exakt festgelegt werden können. So kann bspw. eine Aufhellung/Abdunklung eines Bildes dafür sorgen, dass evtl. für eine bestimmte Bildaussage wichtige Detailbereiche nicht mehr erkennbar sind. Weiterhin kann zu starke Kompression Artefakte erzeugen, wodurch der Bildinhalt nicht eindeutig ist.

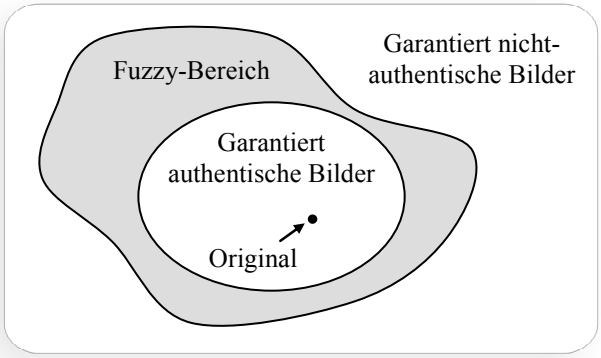


Abbildung 3.10 Unscharfe Trennung zwischen Bereichen authentischer und nicht-authentischer Bilder

Die Definition des Bereiches nicht erlaubter Bildverarbeitungsoperationen hängt von der jeweiligen Anwendung, der Bildgröße und den Eigenschaften (inhaltliche Aussage) des Bildes ab. Zur Simulation nicht erlaubter Operationen wurden in dieser Arbeit die folgenden zwei Ansätze verfolgt:

1. In Form einer einfachen Attacke wurden zufällig ausgewählte 4x4-, 8x8- bzw. 16x16-Pixelbereiche innerhalb der per Wasserzeichen geschützten Bilder vertauscht. Die Bilder wurden anschließend auf Echtheit überprüft. Ziel war die vollständige Ablehnung dieser Bilder (es sei denn, die jeweils ausgetauschten Blöcke sind einander ähnlich).
2. In einem weiteren Test wurden Bilder, in denen kein Wasserzeichen eingebettet ist, auf Echtheit geprüft. Das Ziel, die vollständige Ablehnung dieser Bilder, wurde dabei zu 100% erreicht.

Abbildung 3.11 zeigt die Simulationsergebnisse[32] für Ansatz 1., wobei die Anzahl fälschlicherweise korrekter Verifikationen im Verhältnis zur Anzahl der gesamt getesteten Bilder dargestellt ist (FNR = *false negative ratio*). Folgende Eigenschaften werden dabei deutlich:

[32] Ergebnisse gemittelt über die 52 verwendeten Testbilder bei jeweils 100 Simulationsdurchläufen mit unterschiedlichen Wasserzeichensequenzen.

- Je größer die verwendete Quantisierungsschrittweite Δ gewählt wird, desto wahrscheinlicher tritt der Fall ein, dass Manipulationen nicht erkannt werden.
- Je ähnlicher die gegeneinander ausgetauschten Blöcke sind (größeres PSNR), desto unwahrscheinlicher werden die Manipulationen erkannt (vgl. Abbildung 3.11).
- Je größer die Fläche einer Manipulation ist, desto sicherer wird sie erkannt. Eine Manipulation, wie in Abbildung 3.12 dargestellt, wird bspw. stets sicher erkannt.

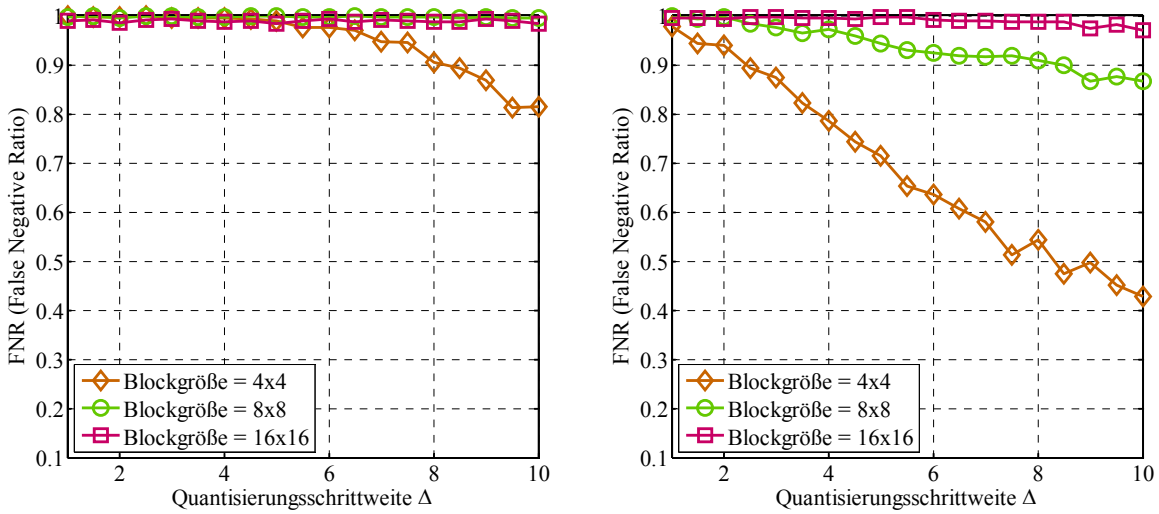


Abbildung 3.11 Simulationsergebnisse der Blockvertauschung (2 Blöcke pro Bild vertauscht) mit einem Unterschied der beiden ausgetauschten Blöcke von PSNR < 15 dB (links), PSNR < 25 dB (rechts)

Abbildung 3.12 Beispiel einer stets sicher erkannten Manipulation: Geschütztes Bild, Δ = 3 (links), manipuliertes Bild - Objekt der Größe 6x9 Pixel entfernt (mittig), kontrastverstärktes Differenzbild (rechts).

3.3.4 Wahrnehmbarkeit der Wasserzeicheneinbettung

Die Wahl der Einbettungsstärke Δ und infolgedessen die Robustheit der eingebetteten Daten ist durch die Wahrnehmbarkeit der resultierenden Bildverzerrungen begrenzt. In Abbildung 3.13 ist ein mithilfe des neu vorgestellten Authentifizierungssystems geschütztes Bild dargestellt, wobei zur Quantisierung der DWT-Koeffizienten die Schrittweite Δ = 8 eingesetzt wur-

de. Vor allem in homogenen Bildregionen sind Artefakte wahrnehmbar[33]. Die Einbettungsstärke für das gesamte Bild darf folglich nur maximal so groß gewählt werden, dass selbst in homogenen Regionen keine störenden Verzerrungen erkennbar sind.

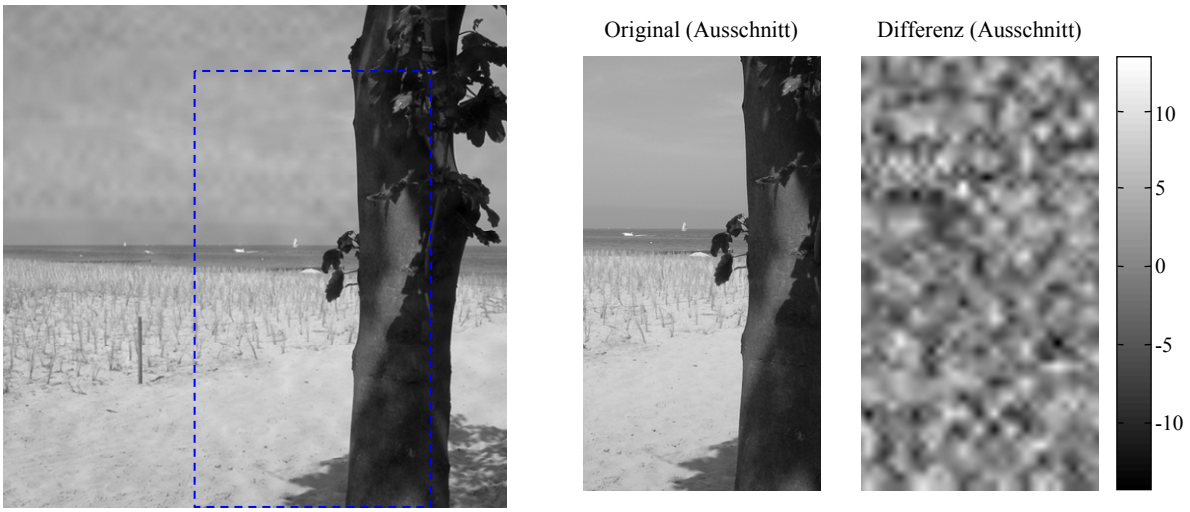


Abbildung 3.13 Beispiel: Bildverzerrungen durch die entwickelte Authentifizierung (Einbettungsstärke $\Delta = 8$)

Zur Erhöhung der Leistungsfähigkeit werden die aus dem entwickelten Authentifizierungsverfahren gewonnenen Erkenntnisse in Kapitel 4 dazu genutzt, das Wasserzeichen an den Bildinhalt angepasst einzubetten. Das Wasserzeichen wird in homogenen Bildregionen mit geringerer und in stärker texturierten Bildregionen mit größerer Quantisierungsschrittweite eingebettet. Es wird gezeigt, dass dadurch die Wahrnehmbarkeit der Bildverzerrungen bei gleichbleibender Robustheit/Kapazität gesenkt werden kann.

3.3.5 Vergleich mit anderen Authentifizierungsverfahren

Für einen Leistungsvergleich des in diesem Kapitel entwickelten JPEG2000-basierten Authentifizierungssystems wurde der von Ekici *et al*. erarbeitete Überblick semi-fragiler Verfahren [ESC+04] herangezogen. In ihrer Veröffentlichung beschreiben die Autoren die Wasserzeichengenerierung, -einbettung und -extraktion von acht der wichtigsten Vertreter dieser Thematik. Zum Vergleich unterziehen sie die Verfahren folgenden Robustheitstests, wobei sie die Einbettungsstärke jeweils derart wählen, dass die markierten Bilder gemessen zum Originalbild ein PSNR von 38 dB bzw. 41 dB aufweisen.

- **Smooth:** Tiefpassfilterung mit 3x3 Filtermaske $[+1\ +1\ +1;\ +1\ +1\ +1;\ +1\ +1\ +1]/9$
- **Sharpen:** Bildschärfung mit 3x3 Unsharp-Filtermaske $[-1\ -1\ -1;\ -1\ +8\ -1;\ -1\ -1\ -1]$

[33] Die Wahrnehmbarkeit der Bildverzerrungen hängt stark von der Darstellung des Bildes ab. Wird das Bild ausgedruckt, sind Artefakte deutlich weniger auffällig als bei der Darstellung an einem PC-Monitor.

- **S and P:** Rauschen *Salt-and-Pepper* (1%) - Grauwerte zufällig auf 0 bzw. 255 gesetzt
- **Histog. equal.:** *Histogram Equalization* (Kontrast-/Helligkeitsänderung des Bildes)
- **AWGN:** Rauschen (Parameter: Standardabweichung $\sigma = 4{,}5$ → PSNR ~ 35 dB)
- **JPEG 70:** verlustbehaftete JPEG-Kompression (Parameter: Quality Factor = 70)
- **Random errors:** zufällige Bit-Fehler mit 0,1% Wahrscheinlichkeit
- **Forgery attack:** Verifizierung des nicht geschützten Originalbildes
- **No attack:** Verifizierung des geschützten, ungestörten Bildes

Die Ergebnisse dieser Tests sind zusammen mit denen des in diesem Abschnitt beschriebenen Authentifizierungssystems (Bezeichnung: Schlauweg) in den Tabellen 3.1 und 3.2 dargestellt[34]. Dabei ist zu erkennen, dass das neu entwickelte System, verglichen mit den anderen Verfahren, oft eine geringere Wahrscheinlichkeit für Fehlentscheidungen besitzt (P_f äquivalent zu *false positive ratio*). Obwohl die entwickelte Authentifizierung, wie in Abschnitt 3.3.2 gezeigt, eine Helligkeits- oder Kontraständerung des Bildes zulässt, ist sie gegenüber der Operation *Histogram Equalization* fragil. Auch das Verrauschen des Bildes mit *Salt-and-Pepper-Noise* führt zur Ablehnung durch den Verifizierungsprozess. Beide Operationen haben deutlich sichtbare Veränderungen des Trägerbildes zur Folge und werden hierdurch im Rahmen dieser Arbeit zur Gruppe der nicht erlaubten Bildoperationen gezählt. Gegenüber der Operation *Sharpen*[35] ist das entwickelte Verfahren ebenfalls nicht vollständig robust.

Semi-fragile method	Forgery attack P_{miss}	Signal-processing attacks P_f							
		No attack	Smooth	Histog. equal.	S and P 1%	AWGN 35 dB	JPEG 70	Sharpen	Random errors
Chang *et al.* [LC00]	0,0 %	0,0 %	100 %	99,0 %	100 %	32,3 %	0,0 %	100 %	0,0 %
Delp *et al.* [LPD00]	0,1 %	2,3 %	54,5 %	3,4 %	6,5 %	2,7 %	2,4 %	0,3 %	14,1 %
Eggers *et al.* [EG01]	0,0 %	0,0 %	41,4 %	91,0 %	2,6 %	0,0 %	0,0 %	65,6 %	2,5 %
Fridrich [Fri99]	1,0 %	1,6 %	62,0 %	5,5 %	19,5 %	2,5 %	25,8 %	21,0 %	2,5 %
Kundur *et al.* [KH99]	0,1 %	0,0 %	77,7 %	99,5 %	51,9 %	10,0 %	2,9 %	98,1 %	0,1 %
Queluz [Que01]	0,01 %	0,01 %	27,8 %	94,3 %	42,7 %	0,01 %	0,01 %	100 %	1,1 %
Liao *et al.* [YLL01]	8,7 %	3,0 %	34,3 %	80,7 %	43,3 %	1,7 %	1,5 %	79,9 %	4,2 %
Tewfik *et al.* [LT99]	N/A	N/A	N/A %	N/A	N/A	N/A	N/A	N/A	N/A
Schlauweg	0,0 %	0,0 %	0,0 %	100 %	100 %	0,0 %	0,0 %	43,7 %	0,0 %

Tabelle 3.1 Leistungsvergleich mit acht anderen Authentifizierungsverfahren (Bezeichnungen und Angaben übernommen aus [ESC+04]) bei einem durch die Einbettung hervorgerufenen PSNR = 41 dB (gemittelt)

[34] N/A = nicht verfügbar (engl. *not available*)

[35] Zur Unsharp-Filtermaske fehlt in [ESC+04] die Angabe eines Wichtungsfaktors. Für die Simulation in dieser Arbeit wurde als Wichtungsfaktor 1/9 angenommen, wodurch eine deutlich sichtbare Bildschärfung erreicht wird. Jedoch sind die Ergebnisse daher nicht direkt mit denen der anderen Verfahren vergleichbar.

Semi-fragile method	Forgery attack P_{miss}	Signal-processing attacks P_f							
		No attack	Smooth	Histog. equal.	S and P 1%	AWGN 35 dB	JPEG 70	Sharpen	Random errors
Chang *et al*. [LC00]	0,0 %	0,0 %	100 %	100 %	100 %	0,0 %	0,0 %	100 %	0,0 %
Delp *et al*. [LPD00]	0,2 %	0,2 %	37,1 %	0,5 %	1,2 %	0,5 %	0,3 %	0,3 %	7,2 %
Eggers *et al*. [EG01]	0,0 %	0,0 %	25,3 %	87,3 %	0,9 %	0,0 %	0,0 %	75,0 %	1,9 %
Fridrich [Fri99]	1,1 %	1,1 %	43,0 %	3,1 %	11,4 %	1,1 %	5,0 %	20,0 %	1,9 %
Kundur *et al*. [KH99]	0,0 %	0,0 %	68,2 %	98,9 %	31,7 %	3,5 %	0,6 %	95,9 %	0,1 %
Queluz [Que01]	0,01 %	0,01 %	15,5 %	87,6 %	7,8 %	0,01 %	0,01 %	99,5 %	1,1 %
Liao *et al*. [YLL01]	16,1 %	0,2 %	12,9 %	59,4 %	2,0 %	0,3 %	0,1 %	57,7 %	0,8 %
Tewfik *et al*. [LT99]	16,9 %	1,4 %	83,5 %	82,7 %	82,6 %	85,5 %	76,1 %	81,4 %	76,8 %
Schlauweg	0,0 %	0,0 %	0,0 %	100 %	100 %	0,0 %	0,0 %	25,5 %	0,0 %

Tabelle 3.2 Leistungsvergleich mit acht anderen Authentifizierungsverfahren (Bezeichnungen und Angaben übernommen aus [ESC+04]) bei einem durch die Einbettung hervorgerufenen PSNR = 38 dB (gemittelt)

Darüber hinaus ist das in dieser Arbeit entwickelte Authentifizierungssystem, wie in Abschnitt 3.3.2 beschrieben, robust gegenüber Skalierungen der Bildgröße, des -kontrastes und der Bildhelligkeit. Zu diesen Bildverarbeitungsoperationen sowie zur Manipulationssicherheit sind jedoch keine Angaben für die Verfahren der anderen Autoren verfügbar.

3.4 Fazit

In den vorhergehenden Abschnitten wurde ein Verfahren zur Authentifizierung von Bildern vorgestellt. Während der JPEG2000-Kompression eines Bildes wird eine durch Quantisierung der DWT-Koeffizienten gebildete Signatur als Digitales Wasserzeichen im Bild eingebettet. Die Einbettung erfolgt ebenfalls mittels Quantisierung, wobei aufgrund der Eigenschaften der DWT-Koeffizienten eine Anpassung der in JPEG2000 zum Einsatz kommenden Quantisierung vorgenommen wurde. Durch die Verifizierung der eingebetteten Wasserzeichen-Signatur kann jederzeit überprüft werden, ob der Bildinhalt verändert wurde.

Das eingebettete Wasserzeichen ist in der Lage, eine Reihe erlaubter Bildsignalverarbeitungsoperationen unbeschadet zu überstehen. Zu den untersuchten, erlaubten Operationen zählen die verlustbehaftete Konvertierung in ein anderes Kompressionsformat, die Veränderung der Bildhelligkeit oder des Kontrastes, das Hinzufügen von Pixelrauschen, die Anwendung eines Glättungsfilters sowie die Skalierung der Bildgröße. Gezielte Angriffe, wie etwa das Hinzufügen, Entfernen oder Retuschieren von Bildregionen, können, wie anhand eigens entwickelter Tests nachgewiesen, erkannt werden. Auf diese Weise ist eine Trennung erlaubter und verbotener Bildoperationen möglich.

Bei der Konzeptionierung, Umsetzung und Optimierung stand stets die praktische Anwendung des Systems im Vordergrund. Es ist gelungen, die einzelnen Komponenten der Authen-

tifizierung effizient zu gestalten, so dass die Gesamtlösung in mobilen Geräten (z.B. Digitalkameras) implementiert werden kann.

Anhand zahlreicher Simulationen wurde die Leistungsfähigkeit des entwickelten Verfahrens untersucht und mit den Daten von Verfahren anderer Autoren verglichen. Dabei hat sich herausgestellt, dass das neue Authentifizierungssystem anderen Verfahren deutlich überlegen ist.

Zur weiteren Verbesserung des vorgestellten Authentifizierungsverfahrens wird im anschließenden Kapitel 4 eine Anpassung der Wasserzeicheneinbettungsstärke an die lokalen Textureigenschaften des Trägerbildes entwickelt. Damit wird das Verfahren gezielt auf die Wahrnehmungseigenschaften des menschlichen Sehens abgestimmt. Diese Weiterentwicklung folgt der im Rahmen dieser Arbeit gewonnenen Erkenntnis, dass Wasserzeichendaten in stark texturierten Bildregionen mit deutlich höherer Stärke eingebettet werden können als in homogenen Regionen. Auf diese Weise kann die Wahrnehmbarkeit der durch die Wasserzeicheneinbettung verursachten Bildverzerrungen bei gleich bleibender Robustheit/Kapazität weiter verringert werden.

Kapitel 4

Bildabhängige Anpassung der Wasserzeicheneinbettung

Im vorhergehenden Kapitel erfolgte die Einbettung des Wasserzeichens unabhängig vom Bildinhalt. Die Wahl der Einbettungsstärke und damit die Robustheit der eingebetteten Daten war durch die Wahrnehmbarkeit der resultierenden Bildverzerrungen, vor allem in homogenen Bildregionen, begrenzt. Als Konsequenz der gewonnenen Erkenntnisse wird in diesem Kapitel eine Anpassung der Wasserzeicheneinbettung an die lokalen Bildtextureigenschaften entwickelt. Durch diese Anpassung an den Bildinhalt sinkt die Wahrnehmbarkeit der Wasserzeicheneinbettung bei gleich bleibender Robustheit/Kapazität. Jedoch kann die zur Anpassung der Einbettung entworfene Bildsegmentierung durch Bildverarbeitungsoperationen beeinflusst werden, wodurch die Leistungsfähigkeit des Systems beeinträchtigt wird. Aus diesem Grund wird die verwendete Fehlerkorrektur auf Seiten der Extraktion durch eine anhand der lokalen Textureigenschaften gewichtete, neu entwickelte Fehlerkorrektur ersetzt. Diese erweiterte Fehlerkorrektur ist für die Steigerung der Gesamtleistungsfähigkeit von großer Bedeutung und bildet daher den Schwerpunkt dieses Kapitels. Abschließend wird das verbesserte Authentifizierungssystem analysiert und mit dem im vorherigen Kapitel entwickelten Verfahren verglichen.

4.1 Motivation

Eine der wichtigsten Forderungen an ein Wasserzeichensystem ist die Nicht-Wahrnehmbarkeit der durch die Einbettung verursachten Trägerbildverzerrungen. Oft werden aus diesem Grund Modelle des menschlichen Sehsystems[36] eingesetzt und die Stärke des Wasserzeichens anhand der Eigenschaften des Bildes ausgewählt.

In der Literatur ist die Anpassung des Wasserzeichens an die Maskierungseigenschaften des menschlichen Sehens unter dem Begriff *perceptual shaping* bekannt [CM02]. Es gibt drei Arten der visuellen Maskierung: die Luminanz-, die Textur- und die Frequenzmaskierung. Jedes Wasserzeichenverfahren, das bspw. nur im niedrigen/mittleren Frequenzband, bzw. nur in ausgewählten Transformationskoeffizienten oder -subbändern einbettet, nutzt bereits die Frequenzmaskierung des Sehens. Das in Kapitel 3 entwickelte Authentifizierungsverfahren bettet das Wasserzeichen nur in ausgewählten Koeffizienten-Subbändern niedriger Frequenz ein und ist somit bereits an die Frequenzmaskierungseigenschaften des Sehens angepasst.

[36] engl. *human visual system* (HVS)

Andere Beispiele finden sich in [CKLS97] oder [LC98]. Die Luminanzmaskierung hingegen erfolgt durch eine Kopplung der Wasserzeichenstärke an die Intensität der Pixelgrauwerte. Ein Überblick derartiger Verfahren findet sich in [WPD99]. Jedoch erst wenige Verfahren (bspw. in [HJJH01], [XZHL06], [LHX01], [TD97]) nutzen die texturmaskierenden Eigenschaften des menschlichen Sehens für die Wasserzeicheneinbettung. Texturmaskierung findet dann statt, wenn Größe und Orientierung des lokal als Störung eingebrachten Wasserzeichens Größe und Orientierung der Textur des Trägerbildes entsprechen (vgl. Abbildung 4.1).

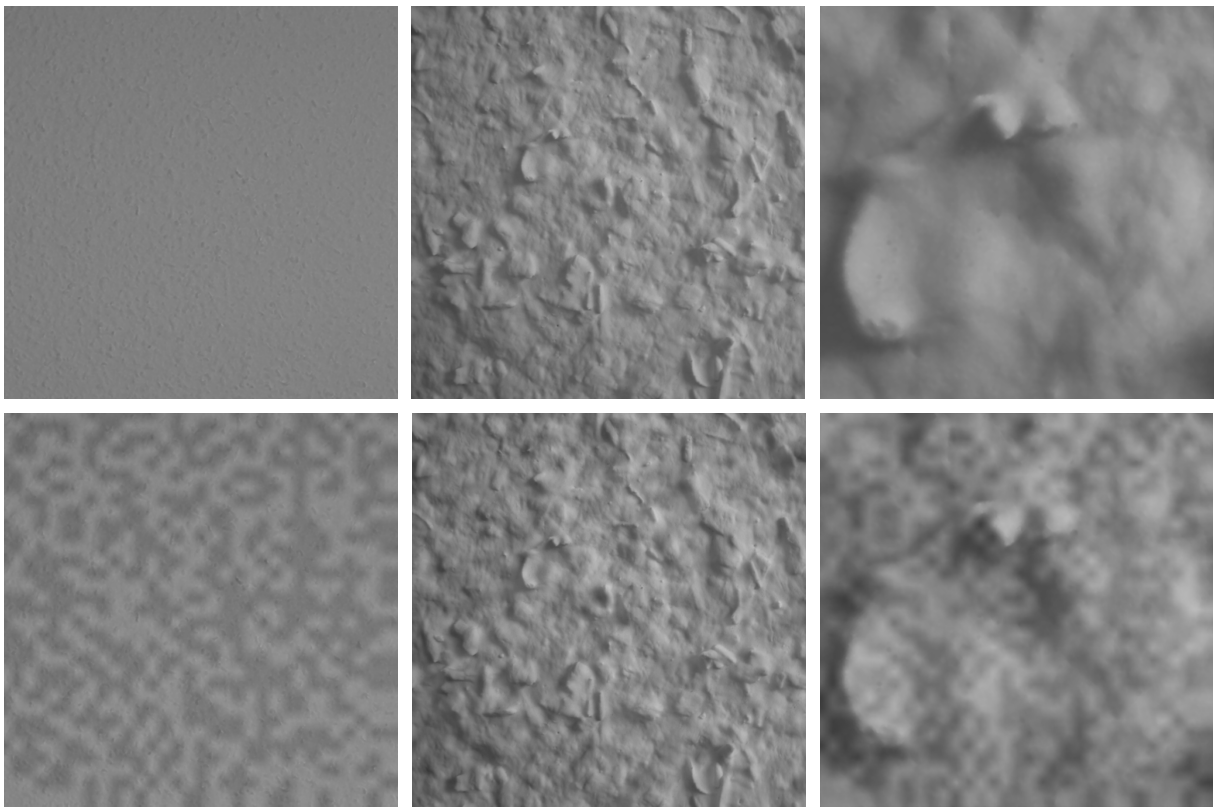


Abbildung 4.1 Beispiel Texturmaskierung: Aufnahmen einer Strukturtapete aus unterschiedlichen Entfernungen (oben), darunter sind die Bilder mit veränderten LL_4-Band-DWT-Koeffizienten gezeigt

Aufgrund der Erkenntnisse des vorhergehenden Kapitels wird das in dieser Arbeit entwickelte Authentifizierungsverfahren in diesem Kapitel um eine Anpassung an die Textureigenschaften des Bildes erweitert. Hierdurch wird eine deutliche Steigerung der Nicht-Wahrnehmbarkeit der Wasserzeicheneinbettung erreicht.

Zunächst werden bekannte Algorithmen vorgestellt und neue Ansätze entwickelt, die das Trägerbild in Bereiche unterschiedlicher Textur aufteilen. Anschließend erfolgt die Wasserzeicheneinbettung in den Teilbereichen mit unterschiedlichen Einbettungsstärken. Dabei wird zwischen einer adaptiven (siehe Abschnitt 4.2) und einer selektiven Einbettung differenziert. Im Gegensatz zur adaptiven werden bei der selektiven Wasserzeicheneinbettung nur ausgewählte (selektierte) Trägerbildbereiche verwendet (siehe z.B. [SPM07b]).

4.2 Texturabhängige Adaption der Einbettungsstärke

In [LHX01] wie auch in [HJJH01] wird die Adaption der Wasserzeicheneinbettung an die Bildtextur jeweils per Schwellwertentscheidung nach Analyse der DWT-Koeffizienten des Bildes vorgenommen. In den Regionen, in denen die berechneten Textureigenschaften einen festgelegten Schwellwert überschreiten, wird eine größere Einbettungsstärke gewählt. In Regionen mit schwächeren Texturmerkmalen kommt zugunsten der geringeren Wahrnehmbarkeit der Einbettung eine kleinere Einbettungsstärke zum Einsatz. Für den Einsatz zweier unterschiedlicher Einbettungsstärken wird das Bild demnach in Bereiche mit geringen und starken Texturmerkmalen unterteilt (vgl. Abbildung 4.2). Eine derartige Unterteilung wird in diesem Kapitel fortan als Einbettungs- bzw. Extraktionsmaske bezeichnet.

(a) (b) (c)

Abbildung 4.2 Bild (a) unterteilt in gering (b) und stark (c) texturierte Bereiche

Bei den Verfahren in [LHX01] und [HJJH01] handelt es sich um korrelationsbasierte Wasserzeichenverfahren (siehe Abschnitt 2.1.1). Anders als in [XZHL06] ist die Einbettungsstärke auf Seiten der Wasserzeichenextraktion irrelevant. In korrelationsbasierten Wasserzeichenverfahren wird das Vorhandensein des (bekannten) Wasserzeichens[37] ausschließlich überprüft (detektiert). Bei quantisierungsbasierten Wasserzeichenverfahren hingegen wird das Wasserzeichen extrahiert. Es ist auf Seiten der Extraktion nicht von vornherein bekannt.

Zur Extraktion des Wasserzeichens muss dieselbe Stärke (Schrittweite) verwendet werden wie beim Einbetten. Es ist wichtig, auf Seiten der Extraktion dieselbe Bildunterteilung zu verwenden. Wird die Einbettungs-/Extraktionsmaske nicht als zusätzliche Hilfsinformation übermittelt, muss sie erneut aus dem Trägerbild berechnet werden.

Unterschiede bei der Bildunterteilung (Segmentierung) führen bei quantisierungsbasierten Wasserzeichenverfahren zu Fehlern bei der Wasserzeichenextraktion (siehe Abschnitt 4.2.2).

[37] Hier spricht man auch von *Zero-Bit-Wasserzeichen*.

Für die Erweiterung der in Kapitel 3 entwickelten, quantisierungsbasierten Authentifizierung werden daher im Rahmen dieser Arbeit folgende Punkte bearbeitet:

- Entwicklung eines Verfahrens zur texturabhängigen Bildunterteilung, welches nach der Anwendung erlaubter Bildverarbeitungsoperationen (vgl. Abschnitt 3.3.2) zu der gleichen Unterteilung führt. Ziel ist die Minimierung der Unterschiede zwischen Einbettungs- und Extraktionsmaske (engl. *mask error rate*).
- Anpassung der Fehlerkorrektur (siehe Abschnitt 4.2.3), um neben dem gestörten, extrahierten Wasserzeichen auch Unterschiede zwischen der Einbettungs- und der Extraktionsmaske korrigieren zu können.

4.2.1 Verfahren zur Textursegmentierung

Nach der Analyse der Segmentierungsalgorithmen aus [LHX01] und [HJJH01] wurden in dieser Arbeit aufgrund der gewonnenen Erkenntnisse zwei weitere Ansätze zur texturbasierten Bildunterteilung entwickelt. Der erste Algorithmus basiert auf der Auswertung der DWT-Koeffizienten des transformierten Bildes. Der zweite legt die Kantendetektion mithilfe des *Sobel-Operators*[38] zugrunde. Durch einen Leistungsvergleich wird derjenige Algorithmus mit der besten Gesamt-Performance ausgewählt.

Textursegmentierung von Huang

Mit Ausnahme des LL-Subbandes besitzen die Koeffizienten eines DWT-zerlegten Bildes in Regionen schwacher Textur geringe Absolutwerte und in stark texturierten Bereichen eine wesentlich größere Amplitude. Auf diese Weise lässt sich ein Bild einfach in Bereiche mit unterschiedlichen Textureigenschaften trennen. So stellen Huang *et al*. in [HJJH01] ein System vor, das die Bereichsunterteilung anhand der DWT-Koeffizienten vornimmt. Dabei werden die Amplituden der einzelnen DWT-Subband-Koeffizienten mit einem Schwellwert verglichen. Die Anzahl der Koeffizienten, die größer als der Schwellwert ist, dient der Ermittlung, ob es sich um einen wenig oder stark texturierten Bereich handelt.

Textursegmentierung von Liu

Einen ähnlichen Ansatz wie in [HJJH01] stellen Liu *et al*. in [LHX01] vor. In ihrem Wasserzeichenverfahren nehmen sie ebenfalls anhand der DWT-Koeffizienten eine Textursegmentierung zur Auswahl der Einbettungsstärke vor. Dabei werden die DWT-Koeffizienten des LL_3-Subbandes in 2x2-Blöcke unterteilt. Die Standardabweichung σ eines jeden Blockes wir mit einem festgelegten Schwellwert verglichen. Ist σ größer als der Schwellwert, wird die Stan-

[38] Der Sobel-Operator ist ein einfaches, in der Bildverarbeitung häufig angewendetes Kantendetektions-Filter.

dardabweichung auch für den dieselbe Pixelregion betreffenden 2x2-Block der HH_3-Koeffizienten berechnet und mit einem Schwellwert verglichen. Diese Prozedur erfolgt ebenso (4x4-blockweise) für die HH_2- und (8x8-blockweise) für die HH_1-Koeffizienten. Erfüllen alle Koeffizienten einer Pixelregion das Schwellwertkriterium, gilt dieser Bereich als stark texturiert.

Neu entwickelte DWT-basierte Textursegmentierung

Die Ansätze von Huang *et al*. und Liu *et al*. wurden im Rahmen dieser Arbeit implementiert und analysiert. Aus den Erkenntnissen wurde ein neues Verfahren zur texturbasierten Bildsegmentierung entwickelt. Dabei werden die Koeffizienten der Subbänder HL_3, LH_3 und HH_3 der dritten DWT-Zerlegungsstufe addiert und mit einem Schwellwert τ verglichen (siehe Abbildung 4.3). Um eine Einbettungs-/Extraktionsmaske in der Größe des LL_4-Bandes zu erhalten, werden die Koeffizienten blockweise im Raster 2x2 gemittelt. Die im Anschluss angewendeten morphologischen Operationen *Closing* und *Erosion* schließen kleine, von stärker texturierten Bereichen umgebene, homogene Bereiche. Die sich ergebende Einbettungs-/Extraktionsmaske erhält die Bezeichnung $F := \{F_n \in \mathbb{R} : 1 \le n \le N\}$. An den Stellen, wo $0 \le F$ ist, wird die größere Einbettungs- bzw. Extraktionsstärke verwendet.

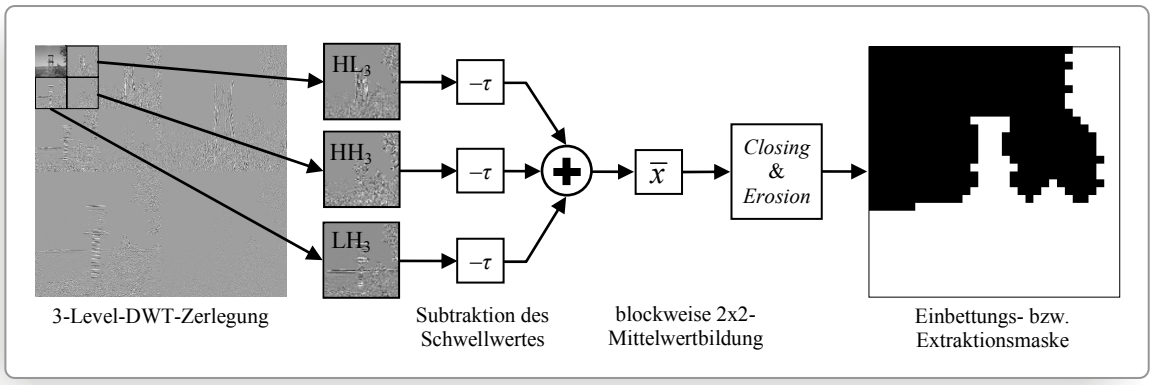


Abbildung 4.3 DWT-basierte neu entwickelte Generierung der Einbettungs-/Extraktionsmaske (τ = 1,5)

Der Schwellwert τ, ab dem ein durch seine DWT-Koeffizienten repräsentierter Bereich als stark texturiert gilt, lässt sich frei wählen. Für die Maske in Abbildung 4.3 wurde τ = 1,5 gewählt. Hierfür ergibt sich bei natürlichen Bildern empirisch die beste Unterteilung.

Neu entwickelte gradientenbasierte Textursegmentierung

Parallel zum neuen DWT-basierten Verfahren wurde in dieser Arbeit ein zweites Verfahren zur Textursegmentierung entwickelt. Es basiert auf der Detektierung von horizontalen, vertikalen und diagonalen Kanten mithilfe eines Gradientenfilters wie dem *Sobel-Operator*. Die resultierenden Kantenbilder werden mit einem Gaußschen Tiefpass geglättet. Anschließend werden diese, wie in Abbildung 4.4 dargestellt, ähnlich dem DWT-basierten Ansatz, zur

Einbettungs-/Extraktionsmaske vereint und mit einem Schwellwert verglichen. Die blockweise 16x16-Mittelwertbildung führt wieder auf die Größe des LL_4-Bandes. Die Operationen *Closing* und *Erosion* dienen, in Analogie zum DWT-basierten Verfahren, der Verfeinerung der Maske an den Übergängen von stark zu gering texturierten Bereichen.

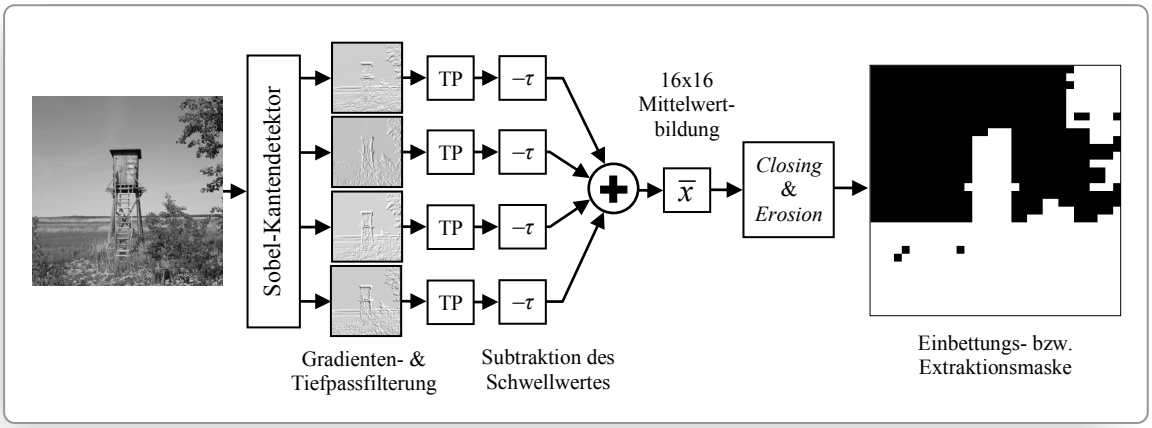


Abbildung 4.4 Gradientenbasierte neu entwickelte Generierung der Einbettungs-/Extraktionsmaske ($\tau = 15$)

Die gradientenbasierten Masken zur Einbettung und Extraktion ähneln denen des DWT-basierten Ansatzes. Für den Schwellwert zeigt sich hier der Wert $\tau = 15$ als geeignet.

Vergleich der Textursegmentierungsverfahren

Die beiden neu entwickelten Verfahren zur Unterteilung eines Bildes in Bereiche mit geringer und starker Textur wurden untereinander und mit den Verfahren von Huang *et al*. [HJJH01] und Liu *et al*. [LHX01] verglichen. Das Ziel ist eine fehlerfreie Reproduktion der Bildunterteilung auf Seiten der Wasserzeichenextraktion auch nach Störungen (vgl. Abschnitt 3.3.2).

Für den in dieser Arbeit verwendeten Satz von Testbildern (siehe Anhang A) wurde die Segmentierungsmaske jeweils für unterschiedliche Schwellwerte τ bestimmt. Danach wurden die Bilder durch die bereits in Abschnitt 3.3.2 aufgeführten Bildverarbeitungsoperationen verändert. Anschließend wurden die Werte der Maske aus den veränderten Bildern erneut berechnet und somit die Wahrscheinlichkeit des Auftretens von Maskenunterschieden MER (*mask error rate*) ermittelt. In Anhang D.1 sind die Testbedingungen und die Simulationsergebnisse detailliert aufgeführt. Abbildung 4.5 zeigt einen Ausschnitt der Ergebnisse.

Anhand der Simulationsergebnisse ist zu erkennen, dass die neu entwickelte gradientenbasierte Segmentierung den anderen Ansätzen überlegen ist. Sie liefert insgesamt die geringste Fehlerwahrscheinlichkeit bei der Reproduktion der Einbettungsmaske nach Störungen. Es sprechen jedoch zwei Gründe dafür, die ebenfalls neue, leistungsstarke DWT-basierte Bildunterteilung für das adaptive Einbetten zu verwenden. Zum einen kann Rechenleistung gespart werden, da die JPEG2000-basierte Authentifizierung samt Signatur-Generierung und

Einbettung ebenfalls in der DWT-Domain stattfindet. Zum anderen lassen sich die Texturmerkmale (DWT-Koeffizienten) für die Segmentierung beim Einbetten des Wasserzeichens gezielt vorverzerren. Auf diese Weise kann, wie im nächsten Abschnitt beschrieben, die Wahrscheinlichkeit für das Auftreten von Maskenunterschieden auf Seiten der Wasserzeichenextraktion effektiv gesenkt werden.

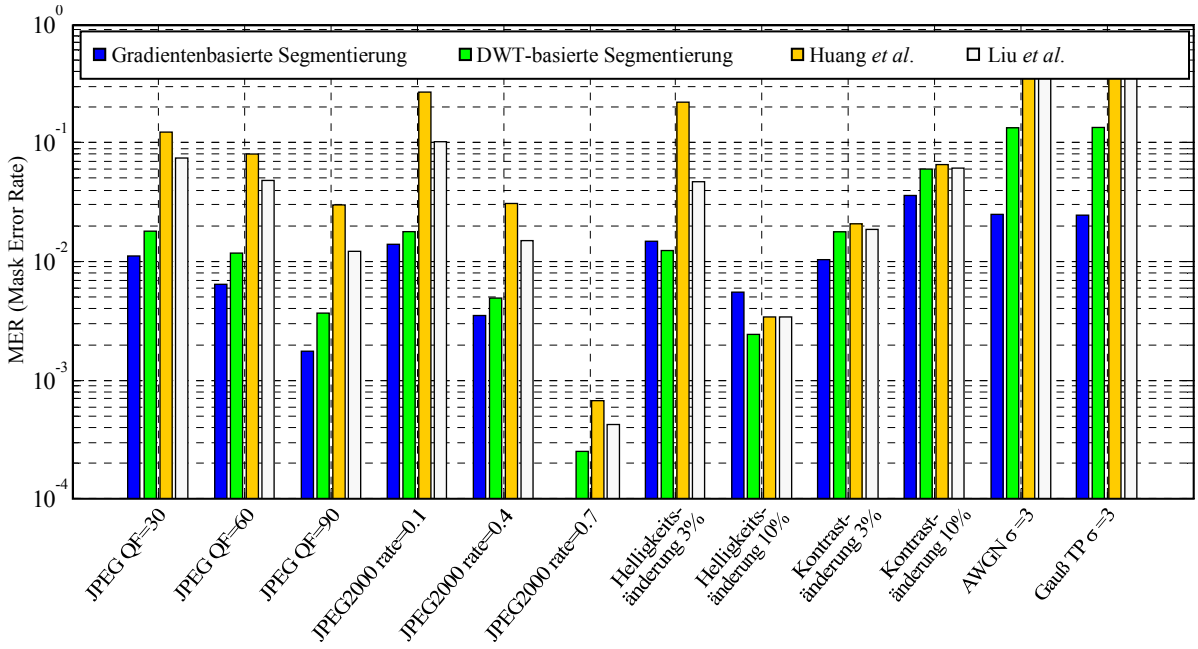


Abbildung 4.5 Robustheit der Textursegmentierungsverfahren gegenüber JPEG-/JPEG2000-Kompression, Helligkeits-/Kontraständerungen sowie Gaußschem Rauschen und Tiefpassfilterung

Vorverzerrung des Segmentierungsmerkmals

Eine oft genutzte Herangehensweise, um Maskenänderungen zu reduzieren, ist die Vorverzerrung des zur Segmentierung genutzten Merkmals F in der Nähe der Entscheidungsschwelle. Dabei wird um den Schwellwert τ ein Toleranzbereich mit der Breite t_τ geschaffen.

$$F = \begin{cases} \tau - \frac{t_\tau}{2} & , \tau - \frac{t_\tau}{2} \le F < \tau \\ \tau + \frac{t_\tau}{2} & , \tau \le F \le \tau + \frac{t_\tau}{2} \\ F & , \text{sonst.} \end{cases} \tag{4.1}$$

Das veränderte Feature muss auf das Trägerbild zurück abgebildet werden. Im Fall der DWT-basierten Segmentierung bedeutet dies, dass die vorverzerrten Koeffizienten während der inversen Diskreten Wavelet-Transformation (IDWT) zu Pixelwerten des Bildes zurück transformiert werden. Für die Wahl der Breite des Toleranzintervalls gilt, umso größer t_τ gewählt wird, desto weniger Maskenänderungen entstehen während der Wasserzeichenextraktion. Jedoch hat ein größeres t_τ auch größeren Einfluss auf die Bildqualität.

Im Anhang D.2 sind die Simulationsergebnisse für eine Vorverzerrung beim DWT-basierten Textursegmentierungsverfahren aufgeführt. Abbildung 4.6 zeigt dazu einen Ausschnitt. Es ist deutlich zu erkennen, dass durch die Vorverzerrung des Texturmerkmals F die Robustheit der Bildsegmentierung um ein Vielfaches gesteigert werden kann.

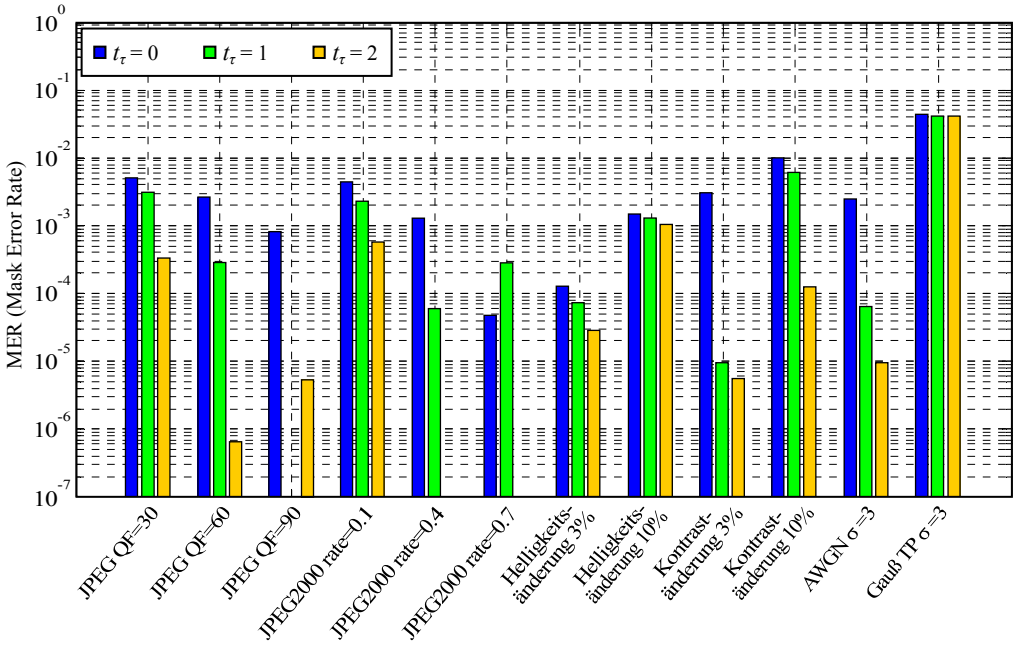


Abbildung 4.6 Robustheit der DWT-basierten Textursegmentierung gegenüber Störungen bei unterschiedlichen Vorverzerrungen des Texturmerkmals

Für den beschriebenen DWT-basierten Segmentierungsansatz ist es leicht, das Texturmerkmal vorzuverzerren, um einen Toleranzgraben um den Schwellwert zu schaffen. Beim gradientenbasierten Ansatz ist eine Vorverzerrung jedoch schwierig. Es kann auch Texturmerkmal-Räume geben, für die keine inverse Transformation bekannt oder nur mit großem Aufwand möglich ist. Dann müssten die Werte der vorverzerrten Segmentierungsmaske empirisch angenähert werden. Darüber hinaus kann bei einigen Wasserzeichenanwendungen die Vorgabe existieren, dass bestimmte Regionen des Trägerbildes nicht verändert werden dürfen. Um die Anzahl der Maskenänderungen trotzdem zu reduzieren, wurde im Rahmen dieser Arbeit neben der Vorverzerrung eine weitere Lösung gefunden (siehe Abschnitt 4.2.3).

4.2.2 Auswirkungen von Fehlern der Bildsegmentierung

Bei den in dieser Arbeit entwickelten Wasserzeichenverfahren wird vorausgesetzt, dass auf Seiten der Extraktion keine Hilfsinformationen[39] für die Segmentierung zur Verfügung stehen (Anforderung des Authentifizierungssystems: *Obliviousness* (siehe Abschnitt 3.1.1)).

[39] Die bei der Einbettung berechnete Segmentierungsmaske wird nicht übertragen oder gespeichert.

Da das Einbetten des Wasserzeichens das Trägerbild bereits verändert (z.B. durch Rundung oder *Clipping*[40]), kann es bei der Reproduktion der Bildunterteilung zu Diskrepanzen zwischen der Einbettungs- und Extraktionsmaske kommen (vgl. Simulationen in Abschnitt 4.2.1).

Steht während des Auslesens der Wasserzeichenbits nicht dieselbe Bildsegmentierung wie beim Einbetten zur Verfügung, werden die Quantisierungsstärken[41] Δ_1 und Δ_2 an den Stellen vertauscht, wo Einbettungs- und Extraktionsmaske nicht übereinstimmen.

Um zu verdeutlichen, wie es durch das Vertauschen der Einbettungsstärken zu Fehlern bei der Extraktion kommen kann, sind in Abbildung 4.7 zwei unterschiedliche Quantisierungsgitter übereinander dargestellt. Das Gitter Λ_1 steht für die Quantisierung mit der Schrittweite Δ_1. Gitter Λ_2 repräsentiert die Quantisierung mit der für dieses Beispiel dreifach größeren Schrittweite Δ_2.

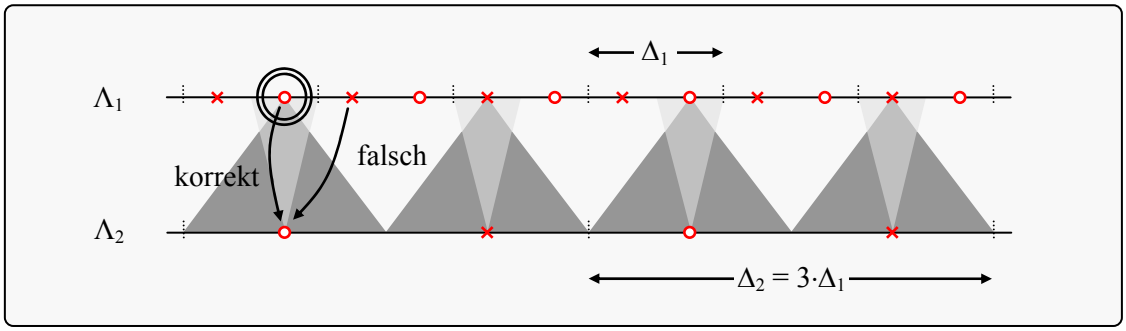


Abbildung 4.7 Entstehung eines Fehlers während der Wasserzeichenextraktion durch das Vertauschen der Quantisierungsstärken Δ_1 und Δ_2

Angenommen, ein Trägersignalwert wurde während des Einbettens als schwach texturiert definiert und im Gitter Λ_1 auf den hervorgehobenen Punkt ⊙ quantisiert. Hat sich anschließend beim Extrahieren des Wasserzeichens die Segmentierung verändert, so dass die betrachtete Position fälschlicherweise als stark texturiert eingestuft wird, kommt es zur Quantisierung in Gitter Λ_2. Da der Punkt ⊙ des Gitters Λ_1 hier auf den Punkt ⊙ in Gitter Λ_2 fällt, der dieselbe Information darstellt, entsteht kein Fehler. Wäre er aber während des Einbettens im Gitter Λ_1 zum Punkt × quantisiert worden, ergäben sich bei der Extraktion Fehler für die von diesem Wert extrahierten Wasserzeichenbits.

Im folgenden Abschnitt wird als Lösung eine Kombination von Textursegmentierung, Wasserzeichenextraktion und Fehlerkorrektur vorgeschlagen und untersucht. Damit können Fehler

[40] Bereichsbegrenzung der Pixelwerte des Bildes, bei 8 Bit Grauwerten $[0;255]$.

[41] Δ_1 wird zum Einbetten in den homogenen und Δ_2 zum Einbetten in den stärker texturierten Bildbereichen eingesetzt.

aufgrund von Maskenänderungen reduziert werden. Ein entscheidender Vorteil ist, dass eine Vorverzerrung des Texturmerkmals auf Seiten der Wasserzeicheneinbettung entfallen kann.

4.2.3 Kombination von Textursegmentierung und Fehlerkorrektur

Ein Schwellwertüber- bzw. -unterschreiten des zur Bildunterteilung verwendeten Merkmals kann, wie im vorhergehenden Abschnitt beschrieben, während der Wasserzeichenextraktion zu Bitfehlern führen. Im Falle selektiver Einbettung, bspw. ausschließlich in den stärker texturierten Bereichen (siehe [SPM07b]), kommt es sogar zu einem Verlust der Synchronisation zwischen Einbettung und Extraktion. Die Folge sind starke Fehlerhäufungen.

Um die Anzahl der Fehler bei der Bildunterteilung zu verringern, wird in dieser Arbeit die Textursegmentierung mit dem Prozess der Fehlerkorrektur vereint (siehe Abbildung 4.8). Die Entscheidung, ob eine Trägersignalposition zum gering oder stark texturierten Bereich gehört, wird in eine weiche Entscheidung[42] überführt und erst während der Fehlerkorrektur im FEC-Decoder getroffen[43]. Dazu wird eine Wichtung der zu decodierenden, extrahierten Wasserzeichendaten $\hat{w}$ über das auf der Extraktionsseite berechnete Segmentierungsmerkmal $\hat{F}$ vorgenommen. Es muss kein spezielles Fehlerkorrekturverfahren verwendet oder bekannte Verfahren verändert werden. Der FEC-Decoder muss lediglich eine Decodierung mit weicher Entscheidung unterstützen, wie bspw. der in dieser Arbeit verwendete Viterbi-Algorithmus für die Decodierung von Faltungscodes.

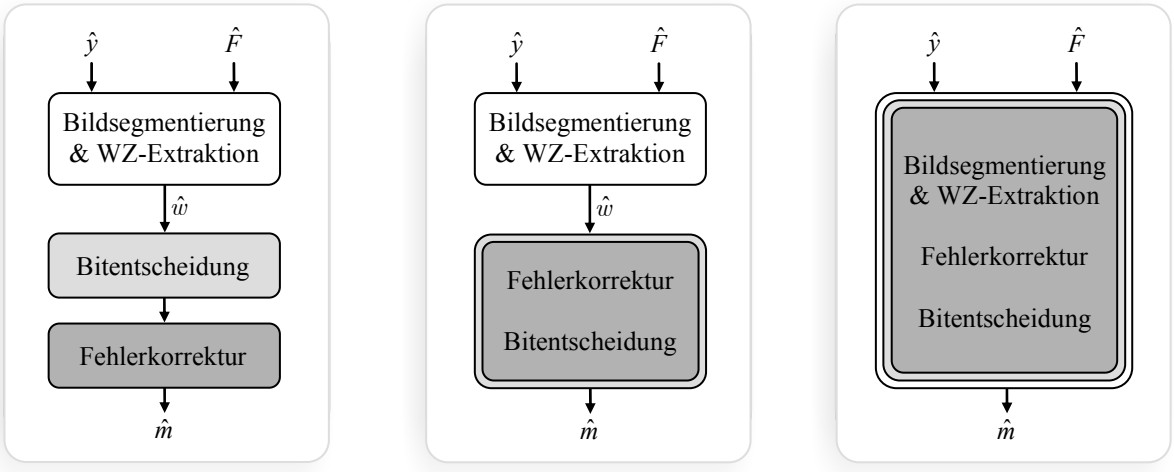


Abbildung 4.8 Kombination von Bildsegmentierung, Wasserzeichenextraktion (kurz: WZ-Extraktion) und Fehlerkorrektur (rechts) als Erweiterung des üblichen Lösungsansatzes einer unabhängigen Bildsegmentierung mit anschließender Bitentscheidung vor (links) bzw. während der Fehlerkorrektur (mittig)

[42] engl. *soft decision decoding*. In der Kanalcodierung bieten einige Fehlerkorrekturverfahren die Möglichkeit des soft decision decoding, wodurch in der Regel eine Verbesserung der Korrekturfähigkeit erreicht wird.

[43] vgl. auch [SPM07a]

Die entwickelte Wichtung basiert auf den zwei im Folgenden erläuterten Eigenschaften:

1. Eigenschaft: Sicherheit der Segmentierungsentscheidung

Bei der Bildsegmentierung erweist es sich als sinnvoll, das Wissen darüber, wie dicht das Segmentierungsmerkmal an der Entscheidungsschwelle τ liegt, für eine Wichtung zu nutzen. Wenn das Merkmal der Schwelle sehr nahe ist, kommt es mit hoher Wahrscheinlichkeit zu einer Fehlentscheidung. Ist das Merkmal für eine Einbettungsposition jedoch weit von der Schwelle entfernt, so ist die Segmentierung (ausgewähltes Quantisierungsgitter) mit hoher Wahrscheinlichkeit korrekt.

Durch Gleichung (4.2) wird in dieser Arbeit die Sicherheit der Segmentierungsentscheidung C als die Distanz des Segmentierungsmerkmals zum Schwellwert definiert[44].

$$C = F - \tau \tag{4.2}$$

Um dieses Wissen für eine weiche Segmentierungsentscheidung zu nutzen, wird das extrahierte Wasserzeichensignal zunächst in zwei Teilsignale aufgespaltet. Dabei sind $\hat{w}_1 = Q_1(\hat{y})$ die im Gitter Λ_1 und $\hat{w}_2 = Q_2(\hat{y})$ die im Gitter Λ_2 quantisierten Werte. Über die Funktionen $f_1(C)$ und $f_2(C)$, mit denen $\hat{w}_1$ und $\hat{w}_2$ nachfolgend multipliziert werden, wird C an das Wasserzeichensignal gekoppelt. Dabei sollen folgende Forderungen erfüllt sein:

- für $C = 0$ sollen $\hat{w}_1$ und $\hat{w}_2$ bei der Wichtung den gleichen Einfluss haben,
- für $C = \pm\infty$ soll das eine Teilsignal bevorzugt und das andere vernachlässigt werden,
- $f_1(C)$ und $f_2(C)$ sollen stetig sein (ohne Sprünge) → weiche Entscheidung.

Abgeleitet von der aus der Nachrichtentechnik bekannten *Raised-Cosine-Function* wurden zu diesem Zweck die in den Gleichungen (4.3) und (4.4) beschriebenen Funktionen entworfen.

Die Flankensteilheit der eingesetzten Cosinus-Funktion lässt sich über den Parameter α, wie in Abbildung 4.9 demonstriert, justieren. Während der Untersuchungen des neu entwickelten Systems führte der Wert $\alpha = 5$ zu den besten Ergebnissen.

$$f_1(C) = \begin{cases} 1 & , -\infty < C < -\alpha \\ \frac{1}{2}\left(1 + \cos\left(\frac{C+\alpha}{\alpha} \cdot \frac{\pi}{2}\right)\right) & , -\alpha \le C < +\alpha \\ 0 & , +\alpha \le C < +\infty \end{cases} \tag{4.3}$$

$$f_2(C) = 1 - f_1(C) \tag{4.4}$$

[44] Für die in Abschnitt 4.2.1 beschriebenen Segmentierungsverfahren sind die Werte der Segmentierungsmaske bereits auf den Schwellwert τ normiert. Dann gilt $C = F$.

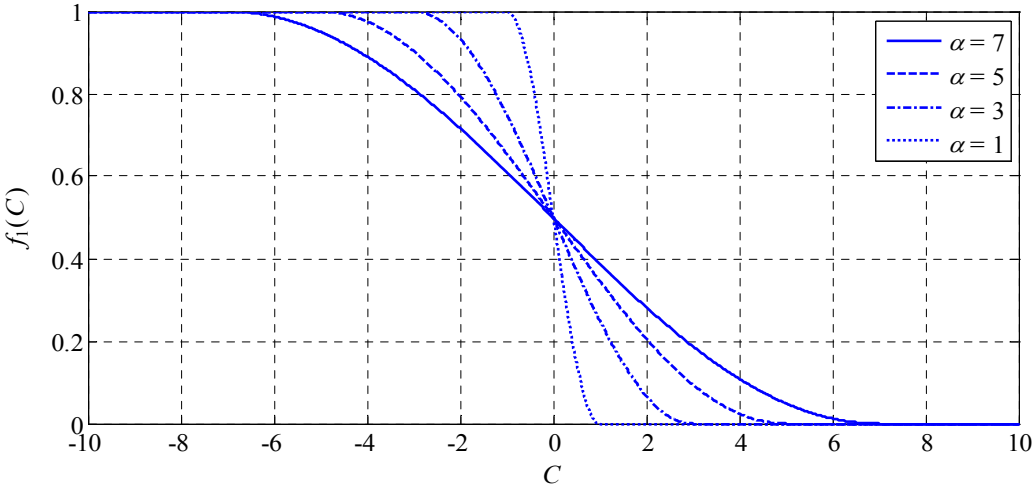


Abbildung 4.9 Wichtungsfunktion $f_1(C)$ bei unterschiedlichen Parametern α für die Flankensteilheit

2. Eigenschaft: Überdeckung der Quantisierungsgitter

Wie bereits in Abbildung 4.7 dargestellt, muss durch das Vertauschen der Quantisierungsschrittweiten Δ_1 und Δ_2 bei der Extraktion nicht zwangsläufig ein Bitfehler entstehen. Bildet das Gitter Λ_2 nämlich ein so genanntes Untergitter von Λ_1 (Gitterpunkte von Λ_2 sind gleichsam Gitterpunkte von Λ_1), dann stellt ein zu diesen Punkten quantisierter Signalwert $\hat{y}$ sowohl im einen als auch im anderen Gitter dasselbe Wasserzeichenbit dar. Eine derartige Überdeckung wird erreicht, wenn Δ_2 / Δ_1 ein ungerades Vielfaches $\left(\Delta_2 = (2n+1)\cdot\Delta_1 : n \in \mathbb{N}^+\right)$ ist.

Bei $\left(\Delta_2 = (4n-1)\cdot\Delta_1 : n \in \mathbb{N}^+\right)$ müssen die Punkte ○ und × des Gitters Λ_2 jedoch, wie in Abbildung 4.10 a) und c) für die Vielfachen 3 und 7 beispielhaft dargestellt, vertauscht wer-

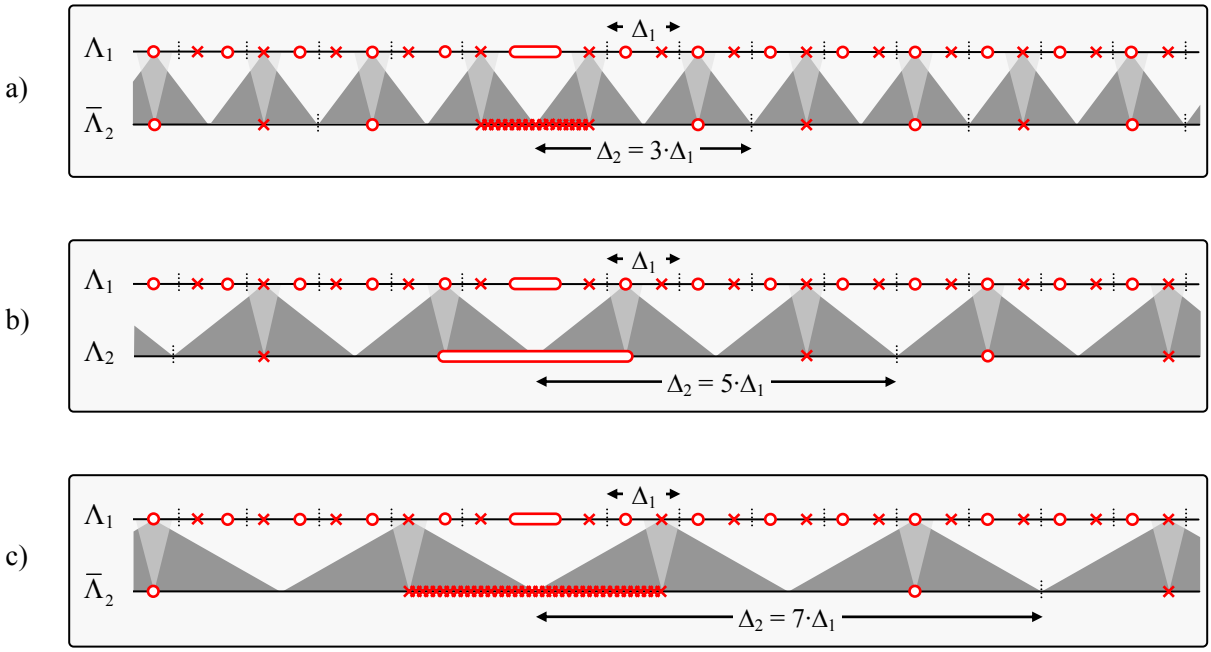


Abbildung 4.10 Überdeckung zweier QIM-Gitter bei ungeraden Vielfachen ihrer Quantisierungsschrittweiten

den. Gitter Λ_2 wird dabei zu $\overline{\Lambda}_2$. Nur dann überdecken dessen Gitterpunkte die Punkte des anderen Untergitters Λ_1. Diese Negierung kommt einer Negierung der über dieses Gitter einzubettenden bzw. extrahierten Wasserzeichenbits gleich und ist somit einfach durchzuführen.

Anhand einer Simulation wurde in dieser Arbeit untersucht, welches Verhältnis von Δ_2 / Δ_1 beim adaptiven Einbetten nach Gleichung (3.1) ohne Fehlerkorrektur die wenigsten Bitfehler ergibt. Dazu wurden 1024 Bit lange zufällige Wasserzeichenbits jeweils in den verwendeten 52 Testbildern adaptiv durch das in dieser Arbeit beschriebene DWT-basierte Wasserzeichenverfahren eingebettet. Nur an Stellen, wo sich aufgrund von Störungen eine Maskenänderung ergab, wurden die Bits ausgelesen und mit der eingebetteten Wasserzeichensequenz verglichen.

Folgende Tabelle zeigt die durch diese Simulation ermittelten Bitfehlerwahrscheinlichkeiten. Es ist deutlich zu erkennen, dass die ungeraden Konstellationen $\Delta_2 / \Delta_1 = 3, 5, 7, \ldots$ aufgrund der Überdeckung der Quantisierungsgitter Λ_1 und Λ_2 am besten für das adaptive Einbetten[45] geeignet sind.

	Δ_2 / Δ_1											
	1,5	2	2,5	3	3,5	4	4,5	5	5,5	6	6,5	7
Λ_2	0,5083	0,4982	0,5012	0,7255	0,4925	0,4938	0,5082	0,2516	0,5061	0,5067	0,4899	0,7231
$\overline{\Lambda}_2$	0,4981	0,5014	0,5062	0,2767	0,5086	0,5098	0,4951	0,7537	0,4963	0,4981	0,5044	0,2811

Tabelle 4.1 Simulierte Bitfehlerwahrscheinlichkeiten aufgrund von Maskenänderungen bei unterschiedlichen Verhältnissen der Quantisierungsschrittweiten Δ_2 / Δ_1 (ohne und mit Negierung des Gitters Λ_2)

Bei den meisten auf das Trägersignal einwirkenden Störungen[46] kann von einem Gauß-verteilten Rauschen (Mittelwert = 0) ausgegangen werden (siehe [VVK+05] bzw. Anhang B.1).

Durch die Quantisierung mit unterschiedlichen Schrittweiten ergeben sich für diese Störungen in den jeweiligen Quantisierungsgittern unterschiedliche Wahrscheinlichkeitsdichteverteilungen. Zur Veranschaulichung der Unterschiede sind in der Abbildung 4.11 (links) die Verteilungen der Störungen und deren Quantisierungsgitter schematisch übereinander dargestellt. Die empirisch anhand einer Simulation mit drei unterschiedlichen Schrittweiten ermittelten Verteilungen befinden sich in Abbildung 4.11 (rechts).

[45] Die Negierung der über Gitter Λ_2 extrahierten Signalwerte im Falle $\Delta_2 / \Delta_1 = 3, 7, 11, 15, \ldots$ wird implizit angewendet und fortan zur Vereinfachung nicht weiter erwähnt.

[46] JPEG- und JPEG2000-Kompression, Helligkeits-/Kontraständerung, Filterung oder additives Pixel-Rauschen.

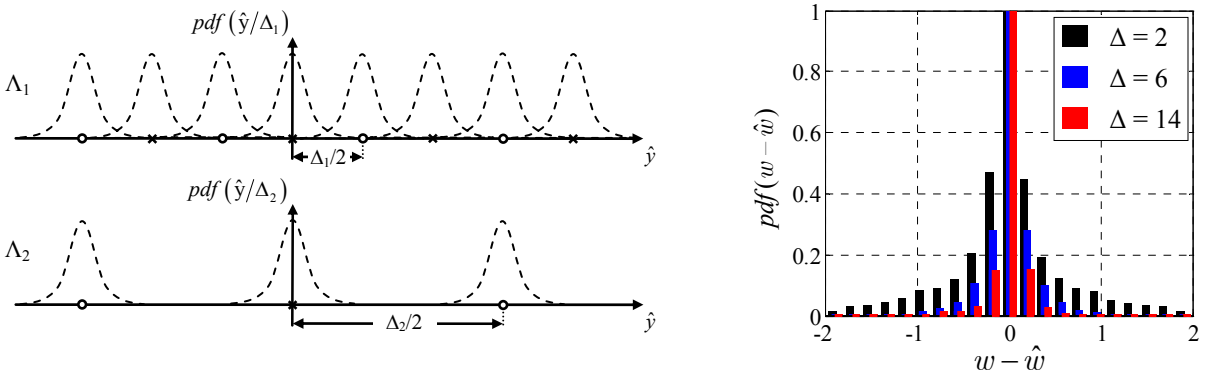


Abbildung 4.11 Wahrscheinlichkeitsdichteverteilung des durch Attacken gestörten Trägersignals. Schematische Darstellung (links), Simulationsergebnisse (rechts)

Es ist zu erkennen, dass es für Gitter Λ_2 Bereiche gibt, in denen sich ein Trägersignalwert deutlich unwahrscheinlicher befindet. Diese Eigenschaft soll ebenfalls bei der Wichtung berücksichtigt werden. Der Algorithmus soll das Teilsignal $\hat{w}_1$ bevorzugen, wenn die Wahl der Quantisierungsgitter bei der Extraktion nicht eindeutig ist und sich der Teilsignalwert $\hat{w}_2$ in Gitter Λ_2 mittig zwischen den Gitterpunkten befindet. Jedoch soll diese Wichtung dann an Effekt verlieren, wenn kein Zweifel an der Segmentierungsentscheidung besteht (wenn z.B. $\alpha < |C|$). Zur Implementierung dieser Eigenschaft wurde die Wichtungsfunktion $f_2(C)$ um den Faktor $\beta(C, \hat{w}_2)$ erweitert und insgesamt noch um das Verhältnis Δ_2 / Δ_1 verstärkt:

$$\beta(C, \hat{w}_2) = 1 - \sin\left(\frac{C+\alpha}{\alpha} \cdot \frac{\pi}{2}\right) \cos\left(\hat{w}_2 \cdot \frac{\pi}{2}\right) \tag{4.5}$$

$$f_2(C, \hat{w}_2) = \frac{\Delta_2}{\Delta_1} \cdot \begin{cases} 0 & , -\infty < C < -\alpha \\ \frac{1}{2}\left(1 - \cos\left(\frac{C+\alpha}{\alpha} \cdot \frac{\pi}{2}\right)\right) \cdot \beta(C, \hat{w}_2) & , -\alpha \le C < +\alpha \\ 1 & , +\alpha \le C < +\infty \end{cases} \tag{4.6}$$

Durch die Wichtung wird $\hat{w}_2$ gegenüber $\hat{w}_1$ im Bereich geringerer Auftretenswahrscheinlichkeit abgeschwächt. Darüber hinaus wird $\hat{w}_2$ zu den Punkten der Gitterüberdeckung hin um den Faktor Δ_2 / Δ_1 verstärkt. Dadurch haben diejenigen extrahierten Wasserzeichensymbole, die mit hoher Sicherheit zu Gitter Λ_2 gehören, während der anschließenden Fehlerkorrektur einen größeren Einfluss. Auf diese Weise können einzelne Bitfehler besser korrigiert werden. Infolgedessen sind die insgesamt eingebetteten Wasserzeichendaten robuster gegenüber Störungen. Die Leistung des Systems wird durch diese neu entwickelte Wichtung, wie im nachfolgenden Abschnitt demonstriert, gesteigert.

Vereinigung der gewichteten Teilsignale und Leistungsanalyse

Nach der Wichtung der beiden Teilsignale werden diese durch Gleichung (4.7) zum Gesamtsignal w^* vereint und einem FEC-Decoder zugeführt. Dieser kann das Signal, wie bereits in

Abschnitt 3.2.2 beschrieben, decodieren. Ein wesentlicher Vorteil der entwickelten Lösung besteht darin, dass weder Veränderungen noch ein spezieller Decoder notwendig sind.

$$w^* = \frac{\hat{w}_1 \cdot f_1(C) + \hat{w}_2 \cdot f_2(C, \hat{w}_2)}{2} \tag{4.7}$$

Die Verbesserung der Wasserzeichenextraktion durch die Kombination von Bildsegmentierung und Fehlerkorrektur wurde anhand umfangreicher Simulationen untersucht (siehe Anhang E.1). Abbildung 4.12 zeigt einen Auszug der ermittelten Bitfehlerverhältnisse für die extrahierten und decodierten Wasserzeichenbits ohne und mit Wichtung (ohne Vorverzerrung des Texturmerkmals). Zum Vergleich (als Referenz) ist die Kurve für die Extraktion unter Kenntnis der Einbettungsmaske dargestellt. Es zeigt sich, dass durch die Wichtung nicht nur die Auswirkung von Maskenänderungen verringert, sondern auch die Fehlerkorrektur allgemein verbessert werden kann. Der Grund dafür liegt im durch die Wichtung herbeigeführten erhöhten Einfluss der über Gitter Λ_2 extrahierten (robusteren) Wasserzeichenbits.

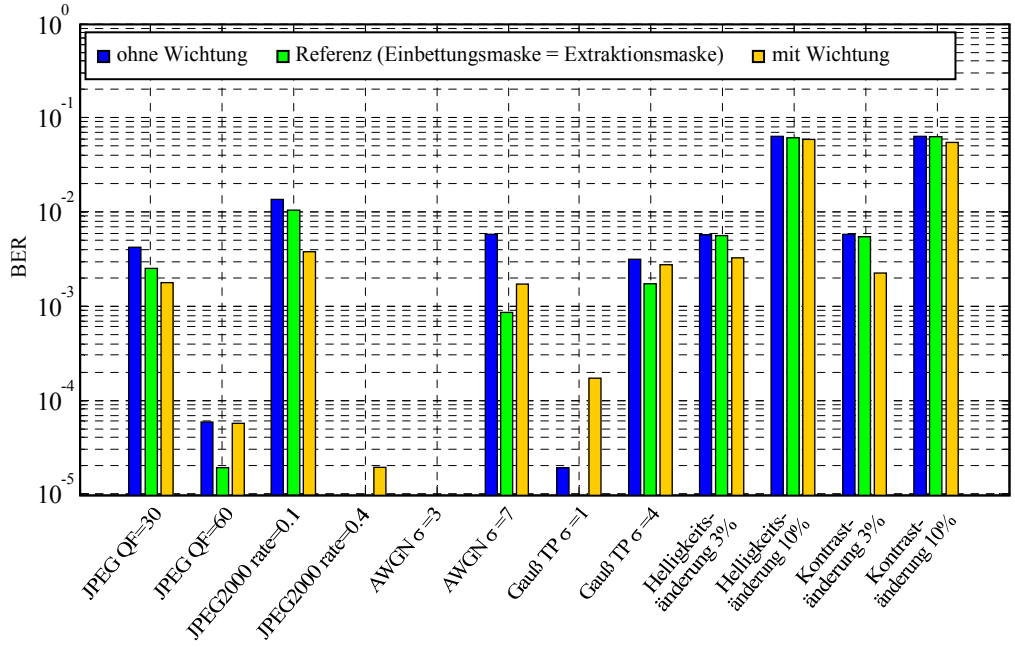


Abbildung 4.12 Robustheit des adaptiven DWT-basierten Wasserzeichenverfahrens (mit und ohne Wichtung) gegenüber Kompression, Helligkeits- und Kontraständerungen, Gaußschem Rauschen und Tiefpassfilterung (Parameter: $\Delta_1 = 3$, $\Delta_2 = 9$, $\tau = 1.5$, $\alpha = 5$)

Die Robustheit des Authentifizierungsgesamtsystems (inkl. Quantisierung der DWT-Koeffizienten zur Hash-Wertberechnung) wird in Abschnitt 4.2.5 untersucht.

4.2.4 Wahrnehmbarkeit der adaptiven Wasserzeicheneinbettung

In den gering texturierten Bereichen werden die DWT-Koeffizienten zur Wasserzeicheneinbettung mit der kleineren Schrittweite Δ_1 quantisiert, anderswo mit Δ_2. Beispielhaft sind in

Abbildung 4.13 das adaptiv mit einem Wasserzeichen versehene Trägerbild sowie das Differenzbild zum Original dargestellt. Weiterhin ist zum Vergleich das Bild mit dem nicht-adaptiv eingebetteten Wasserzeichen gezeigt[47]. Dort wurde an allen Stellen, unabhängig vom Bildinhalt, wie in Kapitel 3 beschrieben, die Schrittweite $\Delta = 6$ verwendet. Man sieht in diesem Bild die störend wirkenden Auffälligkeiten in den homogenen Bereichen, obwohl die PSNR-Werte beider Bilder nahezu gleich sind. Durch das beschriebene adaptive Einbetten ist es gelungen, diese Auffälligkeiten zu vermeiden.

(a) (b) (c)

Abbildung 4.13 Beispiel: (a) Einbettung nicht-adaptiv, unabhängig vom Bildinhalt mit $\Delta = 6$, PSNR = 40,89 dB, (b) Einbettung adaptiv, abhängig von der Bildtextur mit $\Delta_1 = 3$ und $\Delta_2 = 9$, PSNR = 40,98 dB, (c) kontrastverstärktes Differenzbild des mittleren Bildes zum Original

Wie anhand von Abbildung 4.13 gezeigt, ist der PSNR-Wert bei einem an die Maskierungseigenschaften des menschlichen Sehens angepassten Wasserzeichenverfahren nicht geeignet, um die durch die Einbettung verursachten Bildverzerrungen zu beurteilen. Die Veränderungen der DWT-Koeffizienten in den stärker texturierten Bildregionen führen während der PSNR-Berechnung zu einer großen Reduzierung des PSNR-Wertes, obwohl diese dort wesentlich weniger wahrgenommen werden. Aus diesem Grund wird die Wahrnehmbarkeit der adaptiven Wasserzeicheneinbettung in dieser Arbeit fortan nicht mithilfe des PSNR, sondern empirisch (subjektiv) beurteilt. Ausgangspunkt ist dabei die Wahl der Quantisierungsschrittweite Δ_1 für die homogenen Bildregionen. Hierfür hat sich auch bereits während der Untersuchungen der nicht-adaptiven Wasserzeicheneinbettung in Kapitel 3 der Wert $\Delta_1 = 3$ als geeignet erwiesen. Bei dieser Schrittweite sind Veränderungen in homogenen Bildregionen an einem PC-Monitor[48] nicht wahrnehmbar.

[47] vgl. auch Abbildung 3.13

[48] Der PC-Monitor wird im Rahmen dieser Arbeit als Referenz zur Beurteilung der Wahrnehmbarkeit ausgewählt. Der Kontrast ist höher als bei einem Ausdruck der Bilder, wodurch Veränderungen am PC-Monitor kritischer wahrgenommen werden.

Bei den Untersuchungen zur Wahrnehmung der adaptiven Wasserzeicheneinbettung hat sich gezeigt, dass Δ_2 für die Quantisierung in stark texturierten Bereichen durchaus das Fünf- oder Siebenfache von Δ_1 betragen kann. Aufgrund der Texturmaskierung des menschlichen Sehens werden die hervorgerufenen Veränderungen in diesen Bildregionen nicht als störend wahrgenommen. Jedoch an Randbereichen zu homogenen Flächen fallen derart starke Veränderungen auf. Ein Beispiel dazu befindet sich in der folgenden Abbildung 4.14.

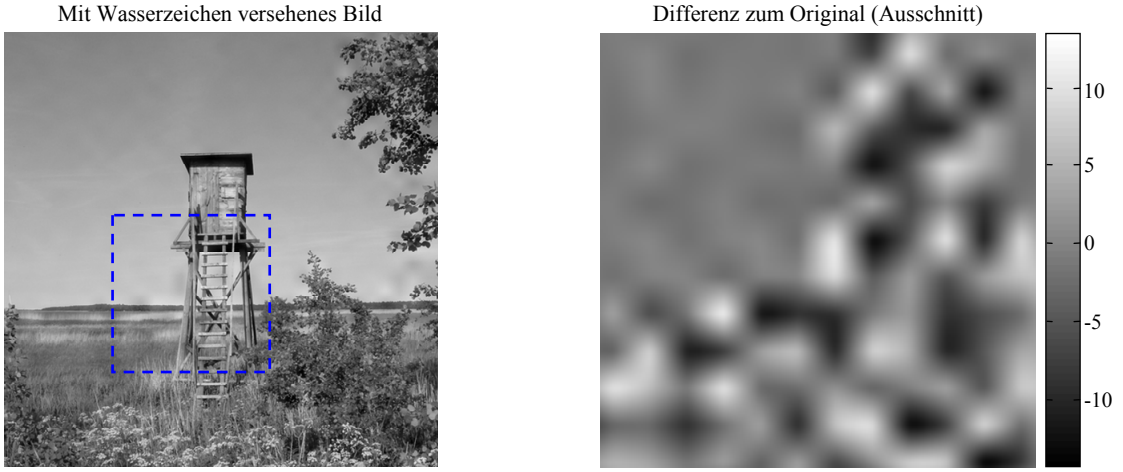


Abbildung 4.14 Beispiel eines im Verhältnis $\Delta_2/\Delta_1 = 9$ durch die adaptive Wasserzeicheneinbettung veränderten Bildes mit $\Delta_1 = 3$ (links); vergrößerter, kontrastverstärkter Ausschnitt der Differenz zum Originalbild (rechts)

Zur Vermeidung der Auffälligkeiten an den Randbereichen dient die Erosion als Teil der in Abschnitt 4.2.1 vorgestellten Segmentierungsverfahren. Wird der dabei verwendete Schwellwert τ erhöht, verringern sich diese Auffälligkeiten. Zwangsläufig verringert sich dadurch jedoch auch die Anzahl der stark texturierten Einbettungspositionen.

4.2.5 Robustheit der adaptiven Wasserzeicheneinbettung

Anhand von Simulationen wurde untersucht, wie sich die Gesamtrobustheit des eingebetteten Wasserzeichens erhöht, wenn das Verhältnis Δ_2 / Δ_1 bei ansonsten gleich bleibenden Parametern vergrößert wird. Abbildung 4.15 zeigt eine Zusammenfassung der im Anhang E.1.2 ausführlich dargestellten Simulationsergebnisse. Anhand dieser Ergebnisse wird auch die Verbesserung deutlich, die insgesamt durch das adaptive Einbetten mit zwei unterschiedlichen Quantisierungsstärken zu verzeichnen ist. Die Kurve für das Verhältnis $\Delta_2 / \Delta_1 = 1$ steht für das nicht-adaptive Einbetten (vgl. Kapitel 3). Es ist zu erkennen, dass bei gleicher Wahrnehmbarkeit der Wasserzeicheneinbettung[49] die Robustheit gegenüber Störungen[50] durch das adaptive Einbetten mit zwei unterschiedlichen Stärken um ein Vielfaches gesteigert werden kann.

[49] gleiches Δ_1

[50] Auswahl von acht bereits in Abschnitt 3.3.2 aufgeführten Bildoperationen

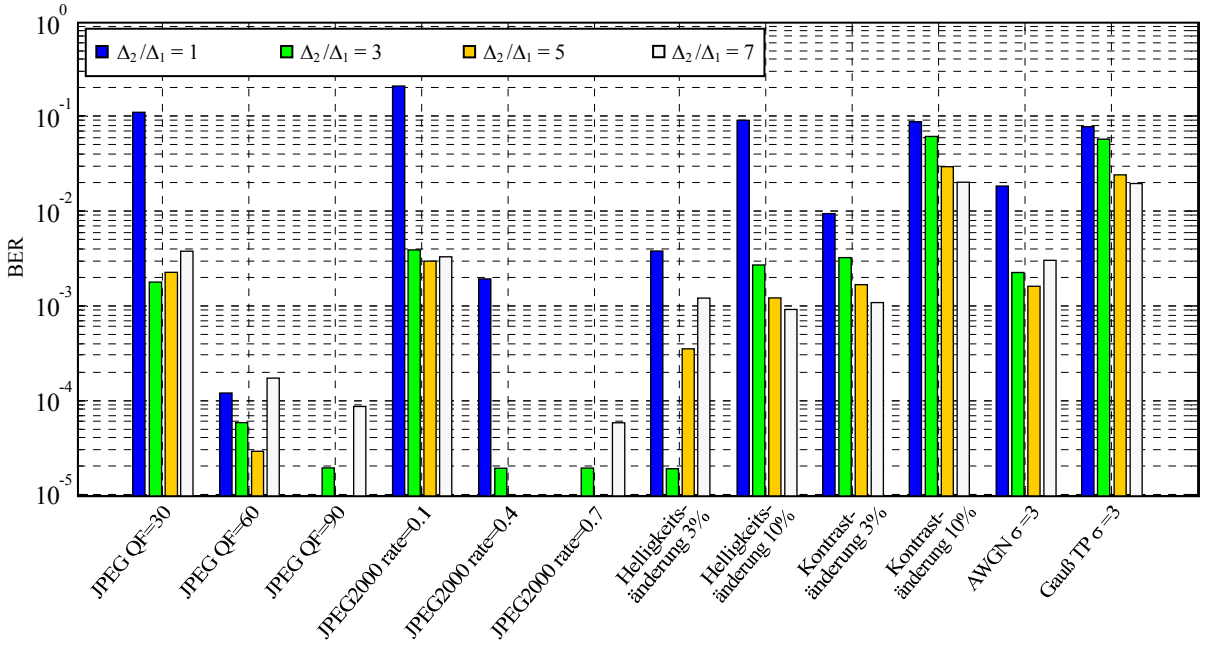


Abbildung 4.15 Robustheit des adaptiven Wasserzeichenverfahrens (mit Wichtung) gegenüber Kompression, Helligkeits- und Kontraständerungen, Gaußschem Rauschen und Tiefpassfilterung bei unterschiedlichen Δ_2 / Δ_1 (Parameter: $\Delta_1 = 3$, $\tau = 1.5$, $\alpha = 5$)

Für die Wahl der beiden Quantisierungsschrittweiten sind die Konstellationen $\Delta_2 / \Delta_1 = 3, 5, 7$ am besten für das adaptive Einbetten geeignet (siehe Abschnitt 4.2.3). Wie Abbildung 4.15 zeigt, wird durch das Verhältnis $\Delta_2 / \Delta_1 = 5$ bzw. 7 gegenüber dem Verhältnis $\Delta_2 / \Delta_1 = 3$ kein wesentlicher Zugewinn an Robustheit erreicht. Aus diesem Grund wird, auch in Hinblick auf die Wahrnehmbarkeit der durch die adaptive Einbettung verursachten Bildverzerrungen (siehe Abschnitt 4.2.4), im Rahmen dieser Arbeit fortan das Verhältnis $\Delta_2 / \Delta_1 = 3$ gewählt.

4.2.6 Leistungsanalyse der erweiterten Authentifizierung

In diesem Abschnitt wird die Robustheit der weiterentwickelten, texturadaptiven Authentifizierung gegenüber erlaubten Bildverarbeitungsoperationen als Ganzes (einschließlich Quantisierung zur Berechnung des bildabhängigen Hash-Wertes) untersucht. Die in Abschnitt 3.3.2 genannten Operationen (Kompression, Tiefpassfilterung, Helligkeits-/Kontraständerung, Bildschärfung sowie Skalierung der Bildgröße) werden auch hier zur Analyse der Leistungsfähigkeit verwendet. Ziel ist es, das Verhältnis fehlgeschlagener Verifikationen gegenüber der Anzahl der gesamt getesteten Bilder (*false positive ratio*) zu minimieren.

Abbildung 4.16 zeigt eine Zusammenfassung der in Abschnitt E.2 des Anhangs ausführlich dargestellten Ergebnisse der Robustheitsanalyse. Es ist zu erkennen, dass die Leistungsfähigkeit der entwickelten Authentifizierung durch die Adaption der Wasserzeicheneinbettung und der Quantisierung zur Hash-Berechnung deutlich gesteigert werden kann. Entscheidend ist,

dass durch diese Weiterentwicklung die Wahrnehmbarkeit der Wasserzeicheneinbettung nicht beeinflusst wird (vgl. Abschnitt 4.2.4).

Die Robustheit der Authentifizierung gegenüber erlaubten Bildverarbeitungsoperationen kann durch die Weiterentwicklung teilweise sogar um das Hundertfache verbessert werden. Der dazu nötige Rechenaufwand kann im Verhältnis zum Aufwand für die gleichzeitig stattfindende JPEG2000-Kompression des Trägerbildes vernachlässigt werden.

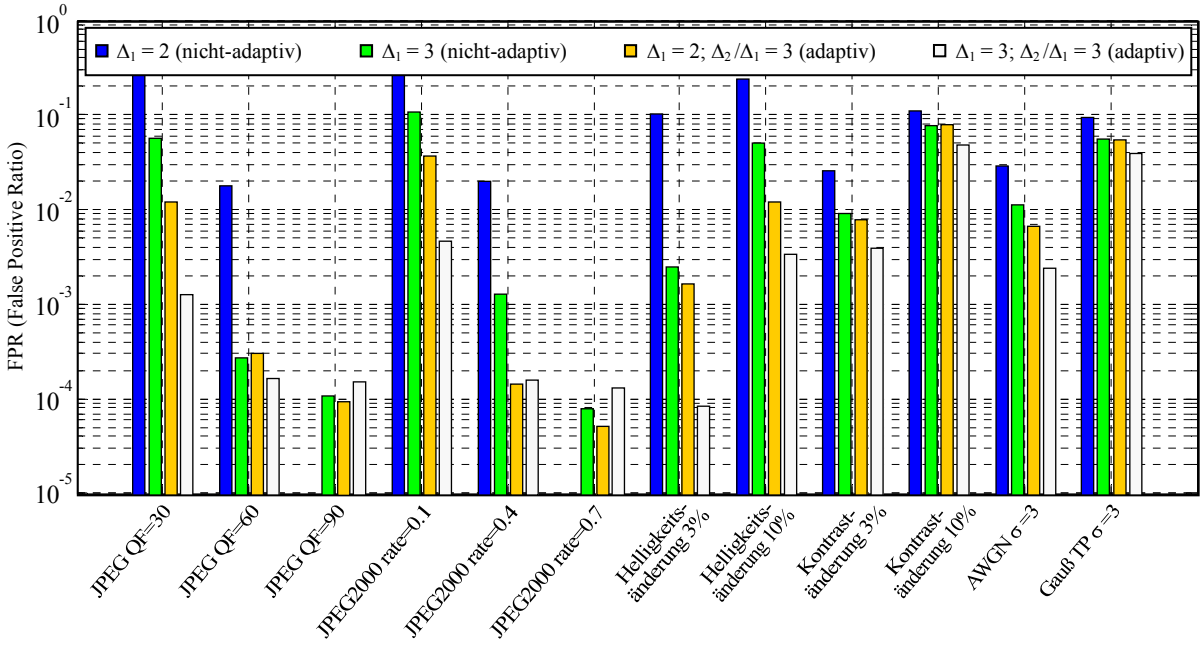


Abbildung 4.16 Robustheit des erweiterten, adaptiven Authentifizierungsverfahrens gegenüber Kompression, Helligkeits- und Kontraständerungen, Gaußschem Rauschen und Tiefpassfilterung (Parameter: $\tau = 1.5$, $\alpha = 5$) im Vergleich mit der nicht-adaptiven Authentifizierung aus Kapitel 3.

Manipulationssicherheit der erweiterten Authentifizierung

Die Verbesserung der Robustheit des eingebetteten Authentifizierungswasserzeichens gegenüber erlaubten Bildoperationen darf nicht zu Lasten der Manipulationssicherheit gehen. Die Verifizierung muss Alarm auslösen, wenn nicht erlaubte Störungen (Veränderungen des Bildinhaltes) angewendet werden.

Um die Manipulationssicherheit des weiterentwickelten, adaptiven Authentifizierungsverfahrens zu ermitteln, wurden die bereits in Abschnitt 3.3.3 aufgeführten Tests angewendet. Das dabei berechnete *false negative ratio* stellt die Anzahl fälschlicherweise korrekter Verifikationen im Verhältnis zur Anzahl der gesamt getesteten Bilder dar. Ziel ist die Maximierung dieses Verhältnisses.

Die in diesem Zusammenhang berechneten Ergebnisse ähneln denen des nicht-adaptiven Verfahrens aus Kapitel 3. So konnte auch für die erweiterte, adaptive Authentifizierung eine hundertprozentige Ablehnung für Bilder ohne eingebettete Wasserzeichen (Test 2) erreicht wer-

den. Die Ergebnisse der Blöcke vertauschenden Attacke (Test 1) sind in Abbildung 4.17 dargestellt. Das geringfügig schlechtere Ergebnis gegenüber dem nicht-adaptiven Verfahren (vgl. Abbildung 3.11) ist auf die größere eingesetzte Quantisierungsschrittweite Δ_2 in den stärker texturierten Bildregionen zurückzuführen. Für gegeneinander getauschte Pixelblöcke ist dadurch in den stärker texturierten Bildregionen ein größerer Toleranzbereich gegeben. Die visuellen Maskierungseigenschaften dieser Bildregionen sorgen allerdings dafür, dass die getauschten Pixelblöcke beim Betrachten nur schwer auffallen. Nichtsdestotrotz werden Manipulationen wie die in Abbildung 3.12 (S. 42) stets sicher erkannt, woraufhin auch die texturadaptive Erweiterung des in Kapitel 3 vorgestellten Authentifizierungsverfahrens als manipulationssicher angesehen werden kann.

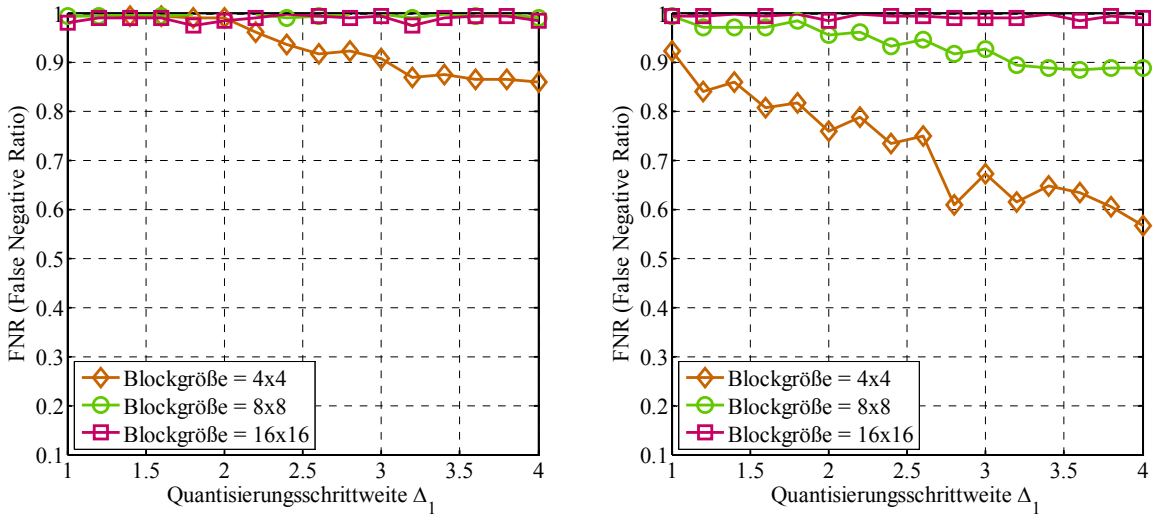


Abbildung 4.17 Simulationsergebnisse der Blockvertauschung (2 Blöcke pro Bild vertauscht) mit einem Unterschied der beiden ausgetauschten Blöcke von PSNR < 15 dB (links), PSNR < 25 dB (rechts); Parameter: $\tau = 1.5$, $\alpha = 5$, $\Delta_2/\Delta_1 = 3$.

4.3 Fazit

In den vorhergehenden Abschnitten wurde das in Kapitel 3 entwickelte Verfahren zur Authentifizierung von Bildern um eine Anpassung an die lokalen Textureigenschaften des Trägerbildes erweitert. In stark texturierten Bildregionen wird eine größere und in nahezu homogenen Bildregionen wird eine kleinere Quantisierungsschrittweite eingesetzt. Damit ist das Verfahren auf die Wahrnehmungseigenschaften des menschlichen Sehens abgestimmt. Auf diese Weise konnte die Robustheit bei gleich bleibender Kapazität und Wahrnehmbarkeit der durch die Wasserzeicheneinbettung verursachten Bildverzerrungen weiter gesteigert werden.

Es wurde gezeigt, dass das eingebettete Wasserzeichen in der Lage ist, eine Reihe erlaubter Bildsignalverarbeitungsoperationen unbeschadet zu überstehen. Zu den untersuchten, erlaubten Operationen zählen die verlustbehaftete Konvertierung in ein anderes Kompressionsformat, die Veränderung der Bildhelligkeit oder des Kontrastes, das Hinzufügen von Pixelrau-

schen, die Anwendung eines Glättungsfilters sowie die Skalierung der Bildgröße. Gezielte Angriffe, wie etwa das Hinzufügen, Entfernen oder Retuschieren von Bildregionen, können, wie anhand eigens entwickelter Tests nachgewiesen, erkannt werden. Auf diese Weise ist eine Trennung erlaubter und verbotener Bildoperationen möglich.

Systembedingt zählen alle Veränderungen der Geometrie des Trägerbildes, wie Rotation, Translation (Verschiebung), Scherung oder das Entfernen von Bildzeilen bzw. -spalten zu den nicht erlaubten Bildverarbeitungsoperationen. Das eingebettete Wasserzeichen und der Hash-Wert können nach der Anwendung dieser Operationen während der Verifikation nicht korrekt berechnet werden.

Bei der Konzeptionierung, Umsetzung und Optimierung stand stets die praktische Anwendung des Systems im Vordergrund. Es ist gelungen, die einzelnen Komponenten der Authentifizierung effizient zu gestalten, so dass die Gesamtlösung in mobilen Geräten (z.B. Digitalkameras) implementiert werden kann.

Anhand zahlreicher Simulationen wurde die Leistungsfähigkeit des weiterentwickelten Verfahrens untersucht und mit den Daten des in Kapitel 3 beschriebenen Verfahrens verglichen. Dabei hat sich herausgestellt, dass durch die Weiterentwicklung deutliche Leistungssteigerungen erreicht werden konnten.

Kapitel 5

Gray-Level-Blob-basierte Wasserzeicheneinbettung

Das in den vorherigen Kapiteln dieser Arbeit entwickelte Wasserzeichenverfahren zur Überprüfung der Echtheit von Bildern weist eine hohe Robustheit der eingebetteten Daten gegenüber den erlaubten Bildverarbeitungsoperationen auf, bei denen die Geometrie des Bildes erhalten bleibt. Wird das Trägerbild jedoch rotiert, verschoben, findet eine Bildscherung statt oder werden wenige Zeilen/Spalten am Bildrand entfernt, dann kann das eingebettete Wasserzeichen nicht korrekt ausgelesen werden. Derartige Operationen treten bspw. auf, wenn ein Bild ausgedruckt und wieder eingescannt wird. In diesem Zusammenhang wurden im Rahmen der vorliegenden Arbeit weitere Untersuchungen durchgeführt. Dabei wurde ein zweites, neues Wasserzeichenverfahren entwickelt, welches aufgrund seiner Funktionsweise zu den so genannten Wasserzeichenverfahren der zweiten Generation zählt. In diesem Kapitel erfolgen eine detaillierte Beschreibung dieses Verfahrens und eine Analyse der Leistungsfähigkeit.

5.1 Motivation

Wird ein Bild ausgedruckt und anschließend eingescannt[51], verändert sich der Bildinhalt nicht wahrnehmbar (siehe Abbildung 5.1). Bildhelligkeit und -kontrast werden geringfügig beeinflusst und es kommt zur Addition einer nicht näher definierten Art von Rauschen.

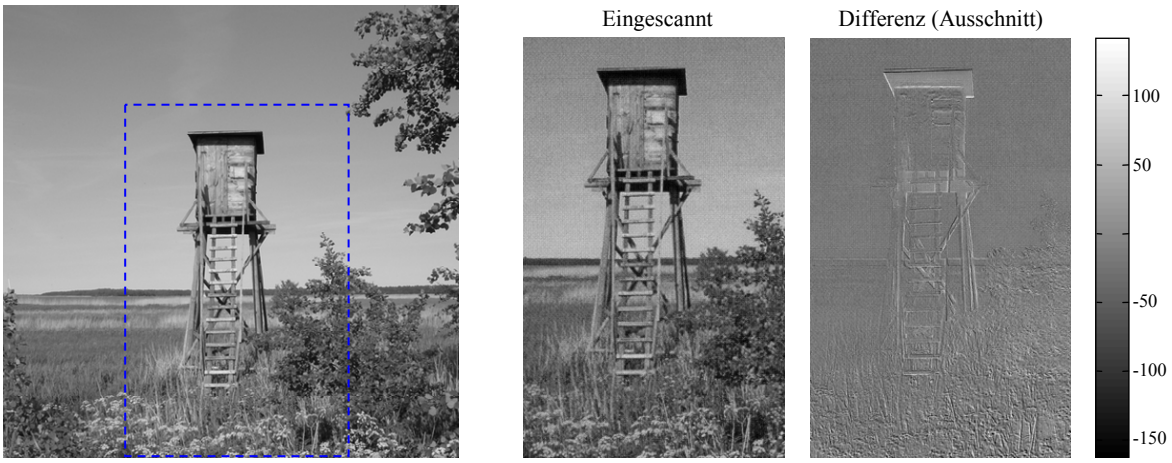


Abbildung 5.1 Original des Beispielbildes “tb24” (links), ein vergrößerter Ausschnitt des Bildes nach dem Ausdrucken und Einscannen (mittig) sowie dessen Differenz zum vergrößerten Originalbildausschnitt.

[51] Für das Ausdrucken der Bilder wurde ein *Hewlett-Packard HP LaserJet 1320n* und für das Einscannen ein *Canon CanoScan Lide 30* verwendet.

Abhängig von der eingesetzten Drucker- und Scanner-Hardware und beeinflusst durch bspw. verkantetes Auflegen eines ausgedruckten Bildes auf den Scanner kann es zu Rotation, Verschiebung, Scherung und/oder Stauchung des Bildes kommen. Darüber hinaus werden oft Bildzeilen und -spalten am Bildrand entfernt (*Cropping*). In Abbildung 5.1 werden diese geometrischen Veränderungen anhand der Differenz des ausgedruckten und wieder eingescannten Bildes zum Originalbild veranschaulicht (vor allem im Bereich von Bildkanten).

Oft wird die Geometrie auch verändert, wenn mit einer Kamera aufgenommene Bilder am Computer nachträglich rotiert oder skaliert werden.

Die durch das in den Kapiteln 3 und 4 entwickelte JPEG2000-basierte Wasserzeichenverfahren eingebetteten Daten können nach Operationen, die die Geometrie des Trägerbildes verändern, nicht extrahiert werden. Durch diese Operationen wird das "Raster der DWT-Koeffizienten", nach dem die Trägersignalpositionen für das Wasserzeichen festgelegt sind, verändert.

Sollen die eingebetteten Wasserzeichendaten gegenüber geometrischen Veränderungen des Trägerbildes robust sein, müsste das in den Kapiteln 3 und 4 vorgestellte Verfahren in zwei Punkten grundlegend verändert werden:

- Vor bzw. während des Generierens und Einbettens der Wasserzeichendaten müsste ein zusätzliches Pilotsignal zur Resynchronisation des Koeffizienten-Rasters im Trägersignal eingebettet oder das Trägerbild geometrisch normiert werden.
- Vor der Extraktion der Wasserzeichendaten müsste das Trägersignal mithilfe des Pilotsignals resynchronisiert bzw. geometrisch normiert werden, so dass die geometrischen Veränderungen des Bildes erkannt und vor dem Extrahieren rückgängig gemacht werden können.

Entgegen dem Versuch, das JPEG2000-basierte Wasserzeichenverfahren in dieser Arbeit zu erweitern, hat es sich als günstiger herausgestellt, ein Verfahren komplett neu zu entwickeln[52].

Der neue Ansatz beruht nicht, wie das erste Verfahren dieser Arbeit, auf der Quantisierung der DWT-Koeffizienten eines Trägerbildes, sondern auf der Detektierung und Veränderung *signifikanter Bildelemente*[53], welche gegenüber geometrischen Bildoperationen robust und nicht an eine feste Rasterunterteilung des Trägersignals gebunden sind.

Verfahren auf der Grundlage signifikanter Bildelemente wurden erstmals 1999 in [KBE99] von Kutter *et al*. als *Wasserzeichenverfahren der zweiten Generation* beschrieben. Der nach-

[52] Ansätze zur geometrischen Normierung mithilfe *geometrischer Momente* (siehe bspw. [DBG+05]) führten im Rahmen dieser Arbeit, für das JPEG2000-basierte Wasserzeichenverfahren nicht zum Erfolg.

[53] umg. Feature-Punkte (engl. *feature points*)

folgende Abschnitt erläutert diesen Begriff, geht auf die damit für ein Wasserzeichensystem verbundenen Eigenschaften ein und gibt einen Überblick zum Stand der Entwicklung.

Das auf der Grundidee von Kutter *et al*. basierende, im Rahmen dieser Arbeit entwickelte, Wasserzeichenverfahren wird im Abschnitt 5.3 beschrieben und hinsichtlich seiner Leistungsfähigkeit analysiert.

5.2 Wasserzeichenverfahren der zweiten Generation

5.2.1 Begriffsdefinition und Eigenschaften

"Im Gegensatz zu den ersten Wasserzeichenverfahren betten die so genannten Verfahren der zweiten Generation Daten in Abhängigkeit vom Bildinhalt ein", so die oft unzureichende Definition. Denn auch die in Kapitel 4 beschriebene trägerbildabhängige Parameteradaption reguliert die Einbettungsstärke in Abhängigkeit vom Bildinhalt. Trotzdem handelt es sich dabei um ein Verfahren der ersten Generation.

Der große Unterschied zwischen Verfahren der ersten und zweiten Generation besteht darin, dass letztere das Trägersignal nicht an einem festen Raster (siehe Abbildung 5.2) ausrichten, egal, ob mit dem Pixel am Bildrand oben links beginnend oder dem ersten Transformationskoeffizienten einer festen Unterteilung des Trägersignals.

Abbildung 5.2 Festes Raster bei Wasserzeichenverfahren der ersten Generation

Um die Robustheit der eingebetteten Daten gegenüber geometrischen Veränderungen des Trägerbildes zu erhöhen, benutzen einige Verfahren der ersten Generation das Originalbild. Mithilfe von Verfahren zur Schätzung und Kompensation verschobener Bildbereiche (bekannt von der Bewegungsschätzung aus der Videocodierung) werden geometrische Veränderungen des Trägerbildes vor der Extraktion der eingebetteten Daten rückgängig gemacht.

Für den Einsatz Digitaler Wasserzeichen zur Bildauthentifizierung kann jedoch kein Originalbild zur Resynchronisation verwendet werden (siehe Abschnitt 3.1.1: *Obliviousness*). Aus diesem Grund ist eine Resynchronisation mithilfe des Originalbildes für das in den Kapiteln 3 und 4 entwickelte Wasserzeichenverfahren nicht geeignet.

Andere Verfahren benutzen so genannte zusätzlich in das Trägersignal eingebettete *Templates* zur Resynchronisation, um das Bild während der Extraktion erneut am Raster auszurichten. Dieses zusätzliche Einbetten von Mustern oder Pilotsignalen hat jedoch eine weitere Reduzierung der Trägerbildqualität bzw. der Einbettungskapazität zur Folge.

Neben den Template-basierten Ansätzen gibt es auch zahlreiche Verfahren der ersten Generation, die das Wasserzeichen in einer Transformations-Domain einbetten, die invariant gegenüber geometrischen Trägerbildveränderungen ist. Diese Methoden nutzen zum Beispiel die *Fourier-Mellin-Transformation* (bzw. *log-polar mapping*), *Zernike-Momente* oder die *Radon-Transformation*. Ein detaillierter Überblick findet sich in [ZLZS07]. Jedoch bleibt Cropping, also das Abschneiden von Bildbereichen, oft ein großes Problem, da auch diese Ansätze das Einbetten über ein festes Raster vornehmen.

Darüber hinaus gibt es Wasserzeichenverfahren, die das Trägerbild oder das Wasserzeichen vor dem Einbetten mithilfe geometrischer Momente normalisieren (siehe bspw. [DBG+05]). Diese Normalisierung ist jedoch für das in dieser Arbeit entwickelte quantisierungsbasierte Wasserzeichenverfahren nicht anwendbar. Die in [DBG+05] beschriebene Normalisierung ist nur für korrelationsbasierte Wasserzeichenverfahren geeignet.

Verfahren der zweiten Generation hingegen, so auch die ursprüngliche Definition von Kutter *et al*. in [KBE99], benutzen signifikante Feature-Punkte des Trägerbildes und richten daran die Wasserzeicheneinbettung bzw. -extraktion aus. Werden Teile des Bildes abgeschnitten oder wird das Bild gedreht, skaliert oder verschoben, so sollen die Feature-Punkte, die noch im veränderten Bild enthalten sind, an denselben Stellen wieder erkannt werden.

In Kutters Definition existieren zwei Ansätze. Einerseits könnte durch die Feature-Punkte ein "am Bildinhalt ausgerichtetes" Raster festgelegt und Daten mithilfe eines Wasserzeichenverfahrens der ersten Generation eingebettet werden. Andererseits könnten die Daten auch "direkt in den Features" untergebracht werden.

Signifikante, inhaltsabhängige Features können Kanten oder Ecken von Bildobjekten sowie Beziehungen zwischen einzelnen Objekten oder Eigenschaften von Objektformen und -texturen sein. Sie sollten zusammenfassend folgende Eigenschaften aufweisen (vgl. [KBE99]):

- Invarianz gegenüber Rauschen
- Kovarianz gegenüber geometrischen Transformationen
- Lokalisierung

Das bedeutet, derartige Features sollten robust sein gegenüber einer verlustbehafteten Kompression des Trägerbildes sowie additivem/multiplikativem Rauschen oder Filterung. Darüber hinaus sollten sowohl globale geometrische Transformationen wie Rotation, Translation und

Skalierung[54], aber auch lokale geometrische Verzerrungen keinen störenden Einfluss auf die Detektierung der Features haben. Lokalisierung bedeutet, dass die verbleibenden Feature-Punkte selbst dann an den originalen Stellen im Bild detektiert werden können, wenn das Trägersignal durch Cropping von Bildzeilen oder -spalten verändert wurde.

5.2.2 State-of-the-Art

In [KBE99] verwenden die Autoren bspw. ein *Mexican-Hat-Wavelet*, um robuste Feature-Punkte im Bild zu finden. Dabei handelt es sich um die Filterung des Bildes mithilfe der normierten zweiten Ableitung einer *Gauß-Funktion*. Die durch die lokalen Maxima der Wavelet-Zerlegung repräsentierten Feature-Punkte spannen *Voronoi-Zellen* auf, in welchen periodisch mithilfe eines Spread-Spectrum-Ansatzes ein Wasserzeichen der ersten Generation eingebettet wird. Mithilfe von *Exhaustive Search* lassen sich einfache geometrische Trägerbildveränderungen erkennen und die eingebetteten Daten zurückgewinnen. Andere Verfahren, die ebenfalls diesen Feature-Detektor verwenden, finden sich in [BCM00], [BCM02], [LSC05] und [TH03]. Das beschriebene Feature zeichnet sich durch Robustheit gegenüber Rotation, Translation, lokalen geometrischen Verzerrungen und in begrenztem Maße gegenüber Rauschen aus. Eine Skalierung des Bildes wird jedoch nicht unterstützt.

Einen anderen Ansatz verfolgen Dittmann *et al*. in [DFS00], indem sie einen *Canny-Kantendetektor* und anschließend einen *Harris-Detektor* benutzen, um im Bild die Ecken von Objekten zu finden. Von diesen Eckpunkten werden Flächen aufgespannt, in denen der Algorithmus Signalmuster zur Resynchronisation geometrischer Störungen einbettet. In [WXQ06] werden kreisrunde Bereiche um die ermittelten Eckpunkte gezogen und zum Einbetten eines ebenfalls auf Korrelation basierenden Wasserzeichens benutzt.

Wieder andere Verfahren, bspw. in [LKL06], verwenden den so genannten *SIFT*-Detektor[55] zur Bestimmung von rotations- und skalierungsinvarianten Regionen für das Einbetten eines Spread-Spectrum-basierten Resynchronisationsmusters. Aber auch eine modifizierte *Hough-Transformation* [LLL07] oder der *Harris-Affine-Detektor* [SY04] kommen zum Einsatz.

Während viele der veröffentlichten Verfahren auf dem ersten Ansatz Kutters [KBE99] beruhen, Features nur zu Zwecken der Resynchronisation zu verwenden, hat die zweite Form, das direkte Einbetten von Daten in die Bild-Features, in der Literatur bislang wenig Aufmerksamkeit erfahren. So schlagen Solachidis *et al*. bspw. vor, Daten in Vektorgrafiken über die Zerlegung dieser mithilfe von *Fourier-Deskriptoren* robust einzubetten [SNP00]. Dabei han-

[54] Rotation, Translation, Skalierung des Trägerbildes (RST)

[55] *Scale-Invariant-Feature-Transform*

delt es sich um eine kompakte Beschreibung geschlossener Objektkonturen, deren Amplitudenspektrum invariant gegenüber Rotation und nach einer Normierung auch gegenüber einer Translation und Skalierung des Bildes ist. Eine andere Idee ist die von Maes und van Overveld [MO98]. Sie betten ein Wasserzeichen ein, indem sie im Pixel-Bereich bestimmte Feature-Punkte auf Gitterpunkte eines festen Rasters verschieben (engl. *warping*). Ihr System ist jedoch nur in der Lage, ein einziges Bit unterzubringen, im Gegensatz zu dem Verfahren von Pröfrock *et al*. [PSM06], das aus dem Bildinhalt so genannte *Pixel Gravity Centers* berechnet und gezielt verändert.

5.3 Wasserzeicheneinbettung anhand von Texturelementen

Im Rahmen dieser Arbeit wurde ein Einbettungsverfahren entwickelt, welches die zweite von Kutter *et al*. [KBE99] formulierte Grundidee verfolgt, Wasserzeichendaten direkt mit den Bild-Features zu verknüpfen.

Den konzeptionellen Überblick dieses neuen, in den folgenden Abschnitten beschriebenen Wasserzeichensystems zeigt Abbildung 5.3.

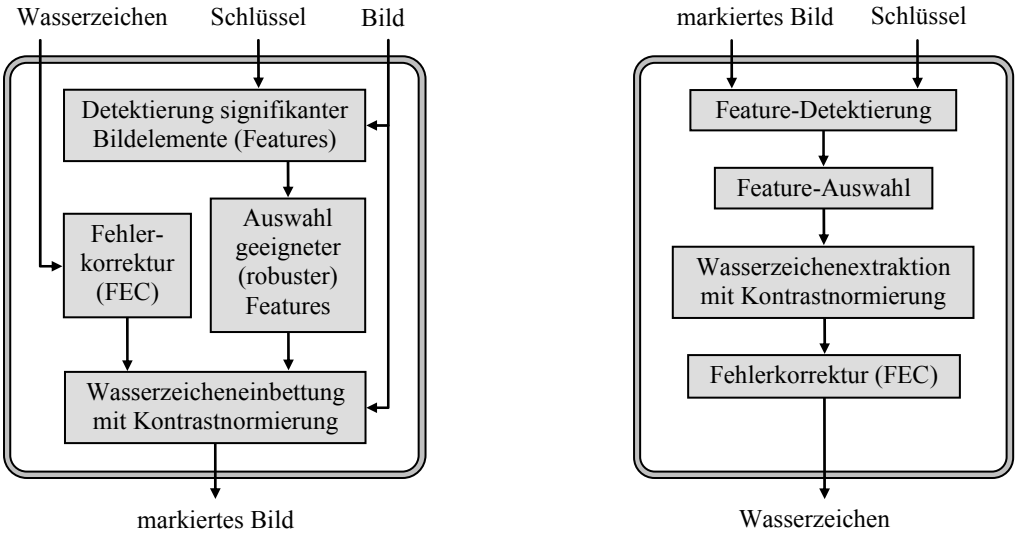


Abbildung 5.3 Entwickeltes Wasserzeichenverfahren auf der Basis signifikanter Bildelemente (Features): Wasserzeicheneinbettung (links) und Wasserzeichenextraktion (rechts)

Als Feature-Detektor wurde eine erweiterte *Mexican-Hat-Wavelet*-Filterung ausgewählt, die in [Lin94] beschrieben und analysiert wurde[56]. In dieser Beschreibung wird davon ausgegangen, dass sich das Bild aus einzelnen Texturelementen (*Texel*) zusammensetzt, die rotations-, translations- und skalierungsinvariant sind. Der Vorteil, direkt mit Texturelementen zu arbei-

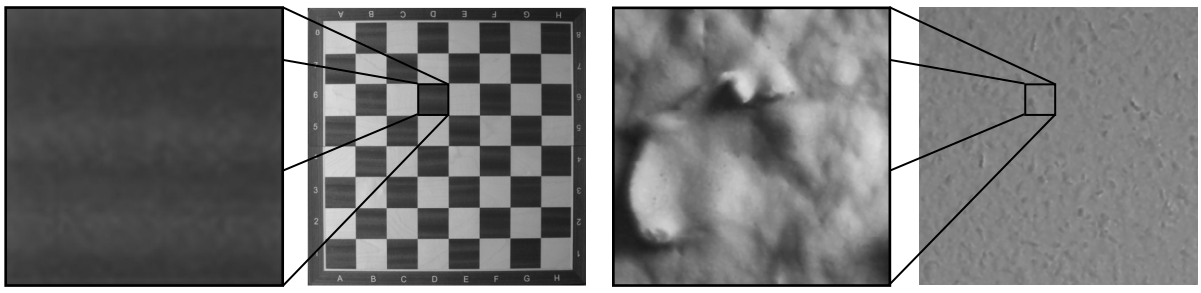
[56] Der in [Lin94] beschriebene Feature-Detektor kommt im Bereich der Gesichtererkennung häufig zum Einsatz. Für das Einbetten von Wasserzeichendaten wird er erstmals im Rahmen dieser Arbeit verwendet.

ten und auf diese Weise die Texturmaskierung des menschlichen Sehens zu nutzen, wurde bereits in Kapitel 4 verdeutlicht. Im Verlauf dieses Kapitels wird gezeigt, dass sich durch das unmittelbare, gezielte Verändern einzelner Texturelemente Wasserzeichendaten mit deutlich erhöhter Robustheit einbetten lassen, ohne dass die Verzerrungen durch die Einbettung wahrnehmbar sind.

Da das ausgewählte Feature im berechneten Feature-Raum (Einbettungs-Domain) je Bildposition durch einen skalaren Wert dargestellt wird, der leicht geändert werden kann, wird zum Einbetten der Daten, wie beim JPEG2000-basierten Wasserzeichenverfahren, eine Quantisierung eingesetzt.

5.3.1 Detektierung von Gray-Level-Blobs im Gaußschen Skalenraum

Wenn die Textur als Feature benutzt wird, stellt die Skalierung einer Texturregion ein Problem dar. Wie am Beispiel in Abbildung 5.4 zu sehen, besitzt eine raue Textur aus der Nähe betrachtet völlig andere Eigenschaften als aus der Ferne. Die Beschreibung der Textur (der Detektor) muss folglich skalierungsinvariant sein.



(a) (b) (c) (d)

Abbildung 5.4 Veränderung der Texturwahrnehmung in unterschiedlichen Betrachtungsabständen: (a) nahe und (b) ferne Ansicht eines Schachbrettes, (c) nahe und (d) ferne Ansicht einer Strukturtapete.

Dieser Problematik widmet sich die *Scale-Space-Theory* (siehe [Lin94], [SY04]), ein Ansatz zur Bildstrukturanalyse, bei dem das Bild in eine Familie von *Scale-Space-Repräsentanten* zerlegt wird. Die charakteristische (optimale) Skale kennzeichnet diejenige Skalierung, bei der der Repräsentant im Skalenraum (Beschreibung der Bildstruktur) einen maximalen Wert annimmt. Der am häufigsten verwendete Skalenraum ist der lineare, *Gaußsche Skalenraum* (siehe Anhang F).

Ein Texturelement im Gaußschen Skalenraum ist der so genannte *Gray-Level-Blob*. Dabei handelt es sich um eine Erhöhung oder Senkung einer Gruppe von Pixeln, ähnlich einer zweidimensionalen Gauß-Glocke. Die in dieser Arbeit eingesetzte Zerlegung eines Bildes in derartige Texturelemente erfolgt durch den in [Lin98] vorgestellten Blob-Detektor, basierend auf einer Transformation des Bildes mit dem *Mexican-Hat-Wavelet*.

Um Gray-Level-Blobs im Bild *I* zu finden, wird dieses im Gaußschen Skalenraum mithilfe von *LOG*-Masken gefiltert. Die Abkürzung *LOG* steht dabei für *Laplacian-of-the-Gaussian*, wobei der auf den Mittelwert Null normierte *LOG-Filter-Kernel* durch das Anwenden des *Laplace-Operators* auf eine zweidimensionale Gauß-Funktion erzeugt wird (siehe Gleichung (5.1)). Der Scale-Space-Parameter σ_r ist die Standardabweichung der Gauß-Funktion.

Das Bild wird nicht nur einmal, sondern für *R* unterschiedliche[57] Skalierungen bzw. Standardabweichungen $\sigma_r = \left\{\sigma_r \in \mathbb{R} : \sigma_{\min} \le \sigma_r \le \sigma_{\max}, r \in \mathbb{N}^+ : r \le R, R \in \mathbb{N}\right\}$ transformiert (siehe Abbildung 5.5). Der Suchbereich für σ_r wird auf $\left[\sigma_{\min}; \sigma_{\max}\right]$ beschränkt, um einen Kompromiss zwischen Kapazität, Robustheit und Rechenaufwand zu ermöglichen. Während der Detektierung (siehe auch Abschnitt 5.3.4) werden diese Skalen an die Bildgröße angepasst[58]. Die resultierenden Matrizen $A(x, y, \sigma_r)$ werden normiert (Gleichung (5.3)), um für die Beträge der einzelnen Filterergebnisse Skalierungsunabhängigkeit zu erreichen (vgl. Anhang F).

$$LOG(x, y, \sigma_r) = \left(\frac{x^2 + y^2}{2 \cdot \pi \cdot \sigma_r^{\,6}} - \frac{1}{\pi \cdot \sigma_r^{\,4}}\right) \cdot e^{-\frac{\left(x^2+y^2\right)}{2 \cdot \sigma_r^{\,2}}} \tag{5.1}$$

$$A(x, y, \sigma_r) = LOG(x, y, \sigma_r) * I(x, y) \tag{5.2}$$

$$A^*(x, y, \sigma_r) = \sigma_r^2 \cdot A(x, y, \sigma_r) \tag{5.3}$$

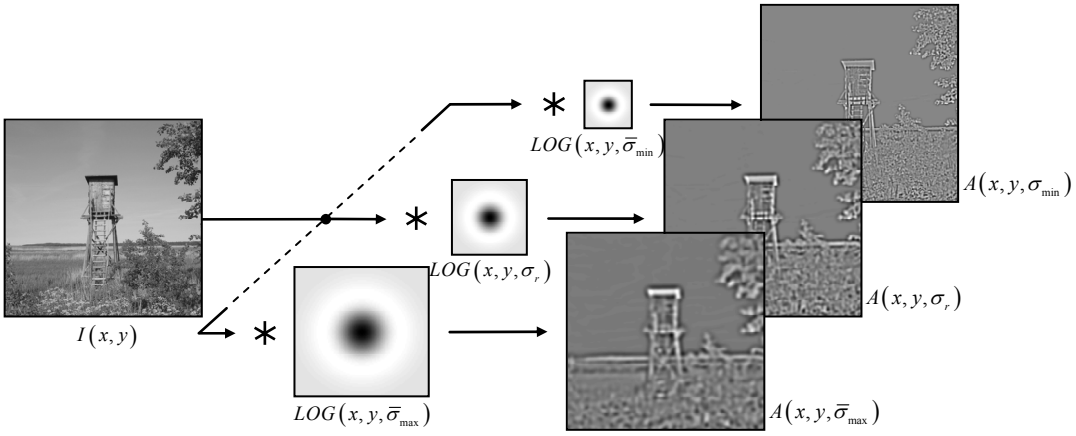


Abbildung 5.5 Bild "tb24" gefiltert mit *LOG*-Filtermasken unterschiedlicher Skalierungen

[57] In den Simulationen wurde *R* = 16 verwendet.

[58] $\bar{\sigma}_{\min} = \sigma_{\min} \cdot \max(X, Y) / 512$ und $\bar{\sigma}_{\max} = \sigma_{\max} \cdot \max(X, Y) / 512$, wobei *X* und *Y* die Dimensionen des Bildes (Bildgröße) darstellen.

Anschließend wird von allen *R* Filterergebnissen für jede Pixelposition (x, y) die *optimale Skale* $\sigma_{opt}(x, y)$ gewählt, die den Term in Gleichung (5.4) maximiert bzw. den größten Betrag $A_{opt}(x, y)$ an der dazugehörigen Stelle aufweist.

$$\sigma_{opt}(x, y) = \arg\max_{\sigma_r} \left| A^*(x, y, \sigma_r) \right| \tag{5.4}$$

Die nachfolgende Abbildung 5.6 zeigt das Ergebnis der Detektierung (Amplitude und optimale Skale) für das Beispielbild "tb24".

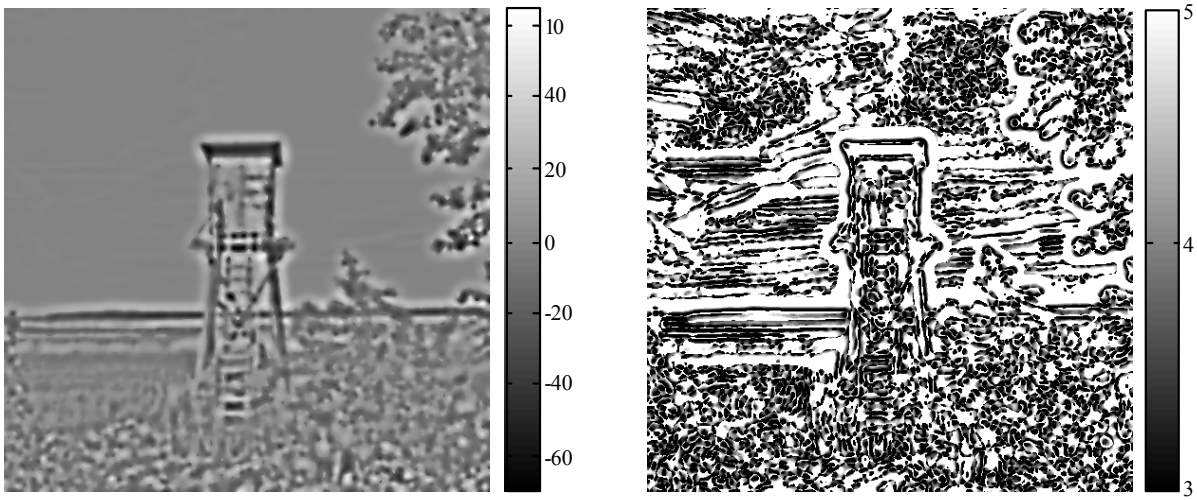


Abbildung 5.6 Ergebnis der *LOG*-Filterung und Auswahl der größten Amplituden A_{opt} (links) und die optimale Skale σ_{opt} (rechts) an den jeweiligen Positionen (x, y)

Die beschriebene Blob-Detektierung ist aufgrund der Eigenschaften der *LOG*-Filterung invariant gegenüber Skalierung, Rotation, Verschiebung sowie horizontaler/vertikaler Spiegelung des Trägerbildes.

Nach dem globalen Verändern der Bildhelligkeit sollten dieselben Gray-Level-Blobs detektiert werden können, es sei denn, durch Clipping werden die Grauwerte des Bildes zu stark verändert.

5.3.2 Wasserzeicheneinbettung in Gray-Level-Blobs

Zur Einbettung von Daten in die Gray-Level-Blobs werden diese quantisiert. Jedoch werden nicht alle im Bild gefundenen Blobs verwendet, sondern nur diejenigen mit den größten Blob-Amplitudenwerten. Sie erwiesen sich während der Untersuchungen im Gegensatz zu Blobs mit kleinen Amplitudenwerten als robuster gegenüber Störungen. Darüber hinaus besitzen sie ausgeprägtere Texturmaskierungseigenschaften.

Die Wasserzeicheneinbettung besteht im Wesentlichen aus den folgenden vier, in den nächsten Unterabschnitten beschriebenen, Schritten:

1. Auswahl der Trägersignalpositionen
2. Verstärkung der Blob-Amplituden
3. Kontrastnormierung der Quantisierungsschrittweite
4. Einbettung der Daten durch Quantisierung der ausgewählten Blob-Amplituden

Auswahl der Trägersignalpositionen

Zuerst wird ein Referenz-Blob B_0 bestimmt, dessen Position für die Reihenfolge verantwortlich ist, mit der die Wasserzeichenbits in die Gray-Level-Blobs eingebettet werden. Als B_0 wird der Wert $A_{opt}(x, y)$ mit der größten Amplitude ausgewählt.

Anschließend werden in einem sukzessiven Prozess K weitere Blobs (Daten-Blobs) ermittelt, bei denen es sich jeweils um die Werte $A_{opt}(x, y)$ mit den größten Blob-Amplituden handelt. Während dieses Prozesses wird um jeden selektierten Blob $B_k := \{B_k(x_k, y_k) : k = 0,1,...,K\}$, wie in Abbildung 5.7 dargestellt, ein Kreis als "reservierter Bereich" markiert, aus dem kein anderer Blob ausgewählt werden kann. Die Größe des Kreises ergibt sich aus der zu dem jeweiligen Blob gehörenden Skale $\sigma_{opt}(x_k, y_k)$. Diese Skale wird mit einem festen Faktor multipliziert, um die *LOG*-Funktion ausreichend genau abzubilden[59]. Dieser Faktor wird in dieser Arbeit als Blob-Maskenmultiplikator O bezeichnet. Er erhält fortan den Wert $O = 6$.

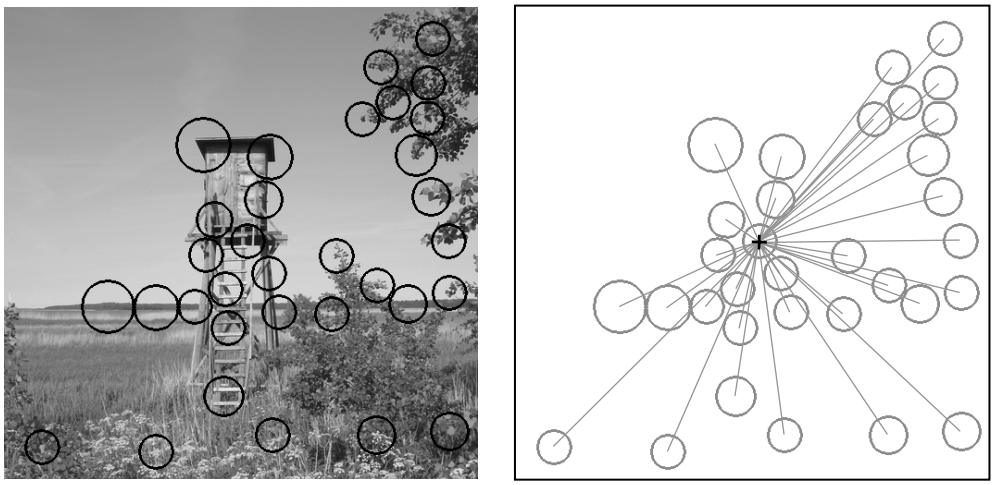


Abbildung 5.7 Beispielbild mit den selektierten $K + 1 = 33$ größten Blobs ($\sigma_{min} = 3$, $\sigma_{max} = 5$)

[59] Die Standardabweichung σ_r einer Gauß-Funktion *Gauß*(x) beschreibt den Wert x auf der Abszisse, für den gilt: $\int_{-\sigma_r}^{\sigma_r} Gauß(x)\,dx = 0,683$. Bei z.B. $\pm 3 \cdot \sigma_r$ beträgt der Wert 0,9999932%. Der Fehler ist dann sehr klein ($6{,}8 \cdot 10^{-6}$).

Es wäre auch möglich, die Überlappung von Blobs zuzulassen. Da sich diese dann beim Einbetten jedoch gegenseitig beeinflussen, wäre ein rekursives (wiederholtes) Einbetten notwendig. Anderenfalls ist eine ausreichende Robustheit nicht sicher gestellt.

Wird keine Überlappung der Blobs zugelassen, gilt für zwei Blobs B_i und B_j der Liste ausgewählter *LOG*-Werte während des Einbettens Gleichung (5.5).

$$\sigma_{opt}\left(x_i, y_i\right) + \sigma_{opt}\left(x_j, y_j\right) \leq d_{ij} \tag{5.5}$$

$$d_{ij} = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2} \tag{5.6}$$

Die Reihenfolge für das Einbetten der Wasserzeichenbits ergibt sich aufsteigend nach der Entfernung d_{ij} der K Daten-Blobs zum Referenz-Blob B_0. In Abbildung 5.7 ist B_0 durch ein Kreuz markiert. Die Entfernungen der Daten-Blobs zum Punkt B_0 berechnen sich nach Gleichung (5.6).

Verstärkung der Blob-Amplituden

Durch Störungen, bspw. aufgrund von Bildverarbeitungsoperationen, aber auch als Folge des Einbettungsprozesses, kann es zu Veränderungen der Werte A_{opt} kommen. Um zu verhindern, dass durch diese Veränderungen auf Seiten der Wasserzeichenextraktion andere als die während des Einbettens ermittelten $K+1$ Gray-Level-Blobs ausgewählt werden, müssen diese mit einer Grundverstärkung G_{base} versehen werden.

Das Einbetten von Wasserzeichenbits in den K Daten-Blobs erfolgt mittels Quantisierung. Dies entspricht einem Verstärken bzw. Abschwächen der Blob-Amplituden um absolute Werte. Grundverstärkung und Dateneinbettung erfordern folglich dieselbe Operation.

Um die Amplitude eines Blobs zu verändern, wird eine *LOG*-Maske $LOG^*_{gain}\left(x, y, \sigma_{gain}\right)$ generiert und zum Blob bzw. zu den Pixelwerten des Bildes addiert. Diese *LOG*-Maske (siehe Abbildung 5.8) kann mithilfe der Gleichungen (5.7) und (5.8) berechnet werden, wobei $G_{diff} \in \mathbb{R}$ die relative Verstärkung zwischen neuer und alter Amplitude darstellt. Die Skale für die *LOG*-Berechnung ist dabei $\sigma_{gain}\left(x, y\right) = \sqrt{2} \cdot \sigma_{opt}\left(x, y\right)$.

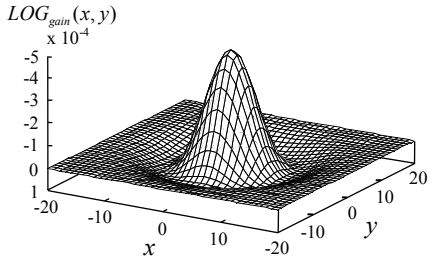


Abbildung 5.8 Beispiel einer *LOG*-Maske ($\sigma_{gain} = 5$)

$$LOG^{*}_{gain}\left(x,y,\sigma_{gain}\right)=\frac{G_{diff}}{K_{norm}}\cdot LOG_{gain}\left(x,y,\sigma_{gain}\right) \tag{5.7}$$

$$K_{norm}=\sigma_{opt}{}^{2}\cdot\sum_{x=-x_{\max}}^{x_{\max}}\sum_{y=-y_{\max}}^{y_{\max}}LOG_{gain}\left(x,y\right)\cdot LOG_{opt}\left(x,y\right) \tag{5.8}$$

Nach der Berechnung der normierten *LOG*-Masken werden diese an den entsprechenden Stellen zu den Grauwerten des Trägerbildes addiert. Folgende Abbildung zeigt ein auf diese Weise verändertes Bild und dessen Differenz zum Original. Die Amplituden der $K+1=33$ ausgewählten Blob-Amplituden wurden dabei alle um den festen Wert G_{base} = 10 vergrößert.

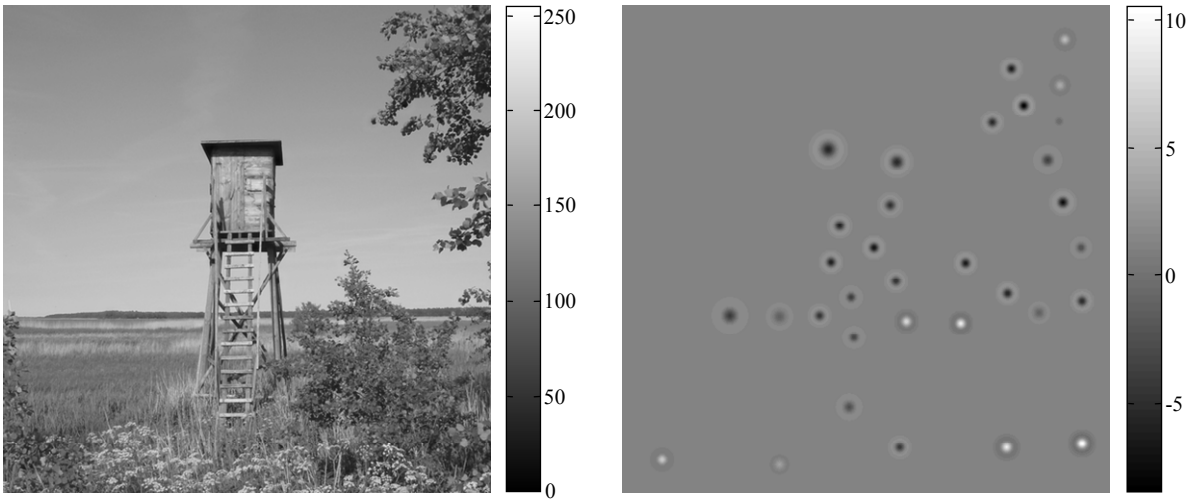


Abbildung 5.9 (links) durch Blob-basiertes Wasserzeichenverfahren verändertes Bild und (rechts) Differenz zum Original (Kontrast für bessere Sichtbarkeit erhöht), K = 32, $\sigma_{\min}$ = 3, $\sigma_{\max}$ = 5, G_{base} = 10, PSNR = 52,986 dB

Einbettung des Wasserzeichens durch Quantisierung der Blob-Amplituden

Zum Einbetten des Wasserzeichens wird, wie bereits beim Verfahren aus Kapitel 3, die *Dither Modulation* (siehe Abschnitt 2.1.1) verwendet. Die Wasserzeichendaten werden durch Quantisierung der ausgewählten Blob-Amplituden $B_k \coloneqq \left\{B_k\left(x_k,y_k\right):k=1,...,K\right\}$ im Bild eingebettet. Die Blob-Amplituden werden jeweils zum nächstgelegenen Quantisierungsgitterpunkt eines von zwei Untergittern $\Lambda_{w_k}=\Delta\mathbb{Z}+w_k\,\Delta/4$ hin verändert, wobei $w_k=\left\{-1,+1\right\}$ das Wasserzeichenbit an der Stelle p_k darstellt. Die Quantisierungsschrittweite beider Untergitter ist Δ.

Kontrastnormierung der Quantisierungsschrittweite

Die *LOG*-Filterung führt zu einer Invarianz der Blob-Amplituden gegenüber Veränderungen der Bildhelligkeit. Eine Veränderung des Bildkontrastes hat jedoch eine proportionale Verringerung/Vergrößerung aller Blob-Amplituden A_{opt} zur Folge. Diese proportionale Beeinflussung der Amplitudenwerte zeigt sich anhand der Simulationsergebnisse in Anhang G.1.

Um für die eingebetteten Wasserzeichendaten ebenfalls Robustheit gegenüber Kontraständerungen zu erreichen, wurde in Analogie zur Normierung in Abschnitt 3.2.1 eine Normierung in Abhängigkeit vom quadratischen Mittelwert entwickelt.

Aus den Werten der Blob-Amplituden wird mithilfe von Gleichung (5.9) ein Normierungsfaktor g berechnet, der mit der Quantisierungsschrittweite Δ verknüpft wird (Gleichung (5.10)). Eine Kontraständerung des Trägerbildes führt zu einer proportionalen Skalierung der Blob-Amplituden. Durch die entwickelte Normierung, die dieser Skalierung entspricht, kann der Wert für die Quantisierungsschrittweite Δ automatisch nachgeführt werden.

$$g = \left(\frac{1}{X \cdot Y} \sum_{x=1}^{X} \sum_{y=1}^{Y} A_{opt}\left(x, y\right)^2 \right)^{1/2} \tag{5.9}$$

$$\Delta = g \cdot P/100 \tag{5.10}$$

Damit hängt die Schrittweite von den Beträgen aller Blob-Amplituden ab. Der Parameter $P \in \mathbb{R}^+$ ist dabei die vom Benutzer wählbare Einbettungsstärke des Wasserzeichenverfahrens.

5.3.3 Eigenschaften von Gray-Level-Blobs

Um die Schrittweite Δ für die Quantisierung zu dimensionieren, wurde während einer Simulation von den verwendeten 52 Testbildern jeweils pro Bild der Blob mit der größten Amplitude $B_0 = \max\left(A_{opt}\right)$ vom Detektor ausgewählt. Die Amplituden der ausgewählten Blobs wurden gespeichert, und die durch unterschiedliche Störungen hervorgerufenen Veränderungen der Amplituden wurden analysiert (siehe Anhang G.1).

Wahrnehmbarkeit der Bildverzerrungen durch die Veränderung eines Gray-Level-Blob

Abbildung 5.10 zeigt exemplarisch einen Ausschnitt des Bildes *Lena*, in dem der Detektor den größten Blob B_0 bestimmt hat (markiert durch den Kreis). Die Amplitude dieses Blobs wird um den Wert $G_{base} = 15$ verstärkt und auf den Wert 85 gerundet. Mithilfe dieser Grundverstärkung ist der Algorithmus in der Lage, denselben Blob auch nach den dargestellten Störungen wieder zu finden.

Die wahrnehmbare Verzerrung, die durch eine Grundverstärkung um den Wert $G_{base} = 15$ entsteht, ist Abbildung 5.10 (b) zu entnehmen. Wie man sieht, ist die visuelle Qualität nicht sichtbar eingeschränkt, obwohl die Pixelwerte im Zentrum der runden Maskierung um den Grauwert 15 verändert werden. Begründet wird dies durch folgende Eigenschaften:

1. Durch die Verstärkung einer Blob-Amplitude entstehen keine Blockartefakte, wie bspw. bei der JPEG-Kompression, weil die Grauwerte der Bildregion, in der sich der Blob befindet, nicht sprunghaft geändert werden (vgl. Abbildung 5.8).

2. Der Feature-Detektor wählt die Blobs mit den größten Amplituden-Werten zum Einbetten des Wasserzeichens aus.

   Da ein großer Amplituden-Wert ausdrückt, dass die Bildregion, in der sich der ausgewählte Blob befindet, bereits sehr gut in Form (Orientierung und Größe) einem Blob (*LOG*) entspricht, wirkt eine starke Texturmaskierung (vgl. Abbildung 4.1, S.48).

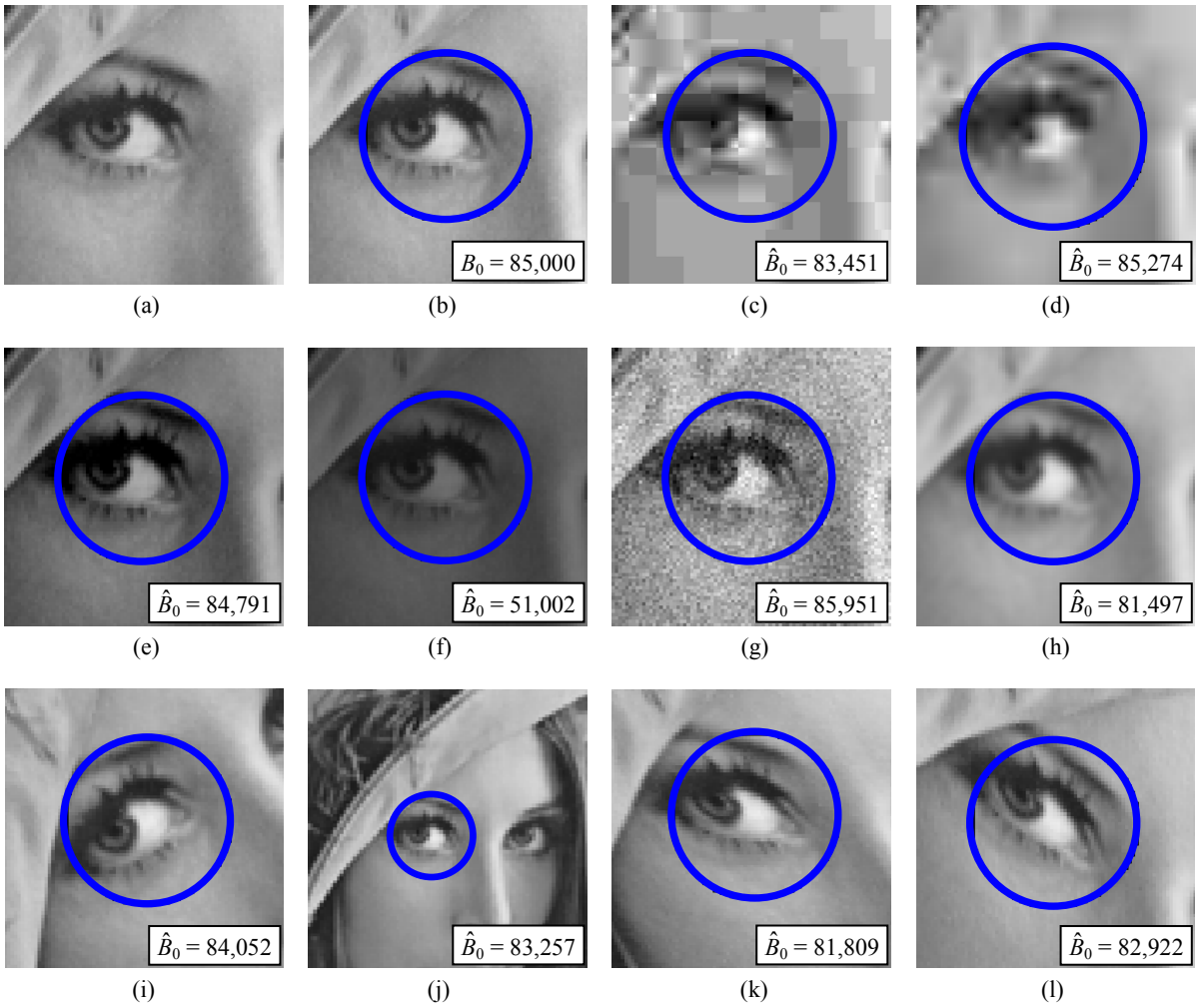


Abbildung 5.10 Bild "tb37" (Ausschnitt) mit größter Blob-Amplitude $\hat{B}_0$. (a) Original, (b) Bild nach Veränderung der Blob-Amplitude um $G_{base} = 15$, verändertes Bild nach (c) JPEG-Kompression mit Qualitätsfaktor $QF = 5$, (d) JPEG2000-Kompression mit Target rate $r = 1/100$, (e) Helligkeitsveränderung, (f) Kontrastveränderung, (g) Gaußschem Rauschen mit $\sigma = 2$, (h) Gaußscher Tiefpassfilterung mit Maskengröße 3x3 und $\sigma = 2$, (i) Rotation um 36°, (j) Skalierung um den Faktor 0.5, (k) horizontaler Scherung um 50%, (l) vertikaler Scherung um 50%

Zur Demonstration der durch die Veränderung eines Blobs verursachten Bildverzerrungen sind in Abbildung 5.11 (a) und Abbildung 5.12 (a) die Bereiche zweier Beispielbilder vergrößert dargestellt, in denen der Detektor jeweils den Blob mit dem größten *LOG*-Amplitudenwert findet. Die übrigen Grafiken dieser Abbildungen zeigen jeweils denselben Bildbereich, nachdem die Blobs an diesen Stellen um den Wert $\Delta = 10$ bzw. $\Delta = 20$ verändert wurden.

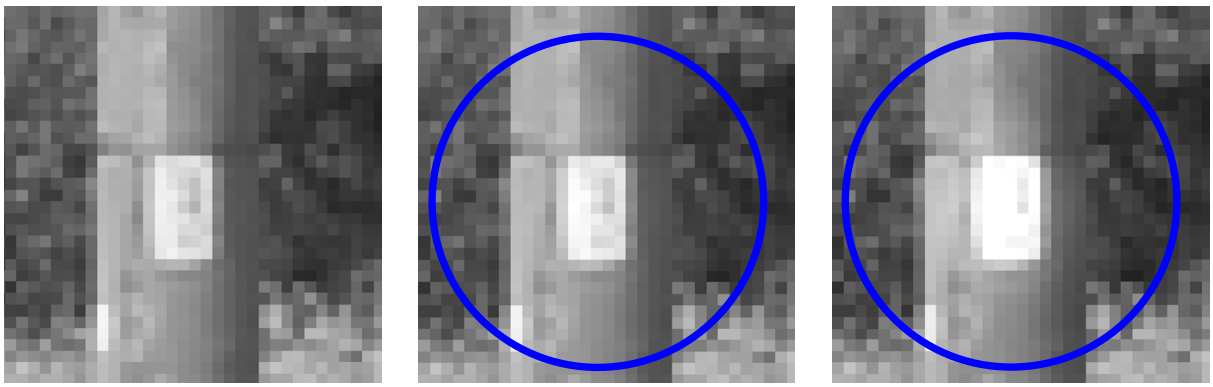


Abbildung 5.11 Beispiele für die Maskierungseigenschaft von Gray-Level-Blobs: 3-fach vergrößerte Ansicht des Originalbildes “tb06” (links) sowie der Bilder, wo der detektierte Blob in der Bildmitte um $\Delta = 10$ (mittig) bzw. um $\Delta = 20$ (rechts) verändert wurde (veränderter Bildbereich markiert durch einen Kreis)

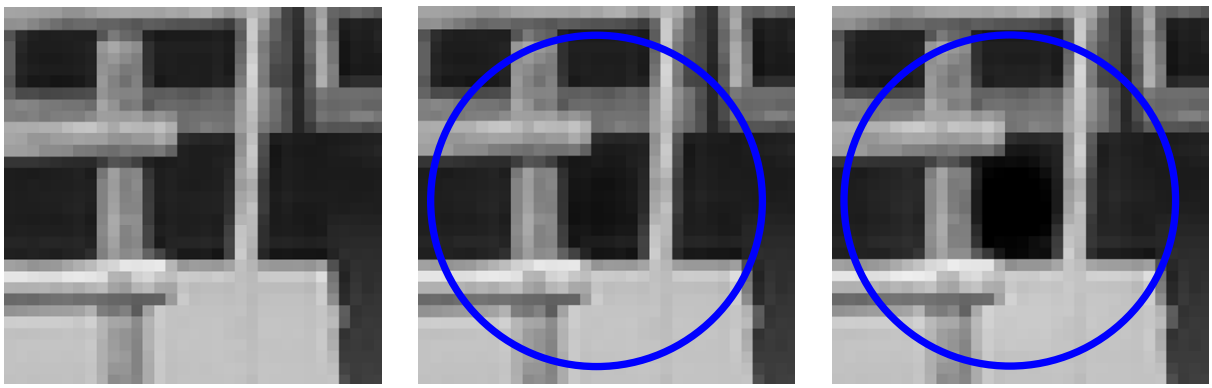


Abbildung 5.12 Beispiele für die Maskierungseigenschaft von Gray-Level-Blobs: 3-fach vergrößerte Ansicht des Originalbildes “tb24” (links) sowie der Bilder, wo der detektierte Blob in der Bildmitte um $\Delta = 10$ (mittig) bzw. um $\Delta = 20$ (rechts) verändert wurde (veränderter Bildbereich markiert durch einen Kreis)

Es ist zu erkennen, dass die Grauwerte, vor allem in der Mitte jeder einzelnen Grafik aus Abbildung 5.11 bzw. Abbildung 5.12, leichte Unterschiede zueinander aufweisen. Wenn man nicht davon ausgeht, dass links die Originalbildbereiche dargestellt sind, wirken die Bildveränderungen nicht unmittelbar störend.

Abgesehen von den durch Clipping verursachten unnatürlich dunkel bzw. hell wirkenden Bildbereichen entstehen durch das Verändern der ausgewählten *LOG*-Amplitudenwerte keine auffälligen Artefakte.

Veränderungen der Blob-Amplituden infolge von Störungen

Die Notwendigkeit einer Blob-Grundverstärkung G_{base} während der Wasserzeicheneinbettung wurde anhand von Simulationen untersucht (siehe Anhang G.1).

Wenn keine Grundverstärkung stattfindet ($G_{base} = 0$) und dadurch in einigen Fällen nicht derselbe Blob im Bild wiedererkannt wird, kommt es zu Ausreißern innerhalb der Simulationsergebnisse. Die Kurven zeigen auch, dass eine Kontraständerung die Größe der Blob-Amplituden proportional beeinflusst. Wird beim Einbetten der Wasserzeichendaten die im

vorherigen Abschnitt beschriebene Anpassung der Quantisierungsschrittweite vorgenommen, kann die Skalierung der Blob-Amplituden durch die Kontraständerung ausgeglichen werden.

Die Kontrastnormierung hat auch im Falle der Gaußschen Tiefpassfilterung oder der Bildschärfung mittels Unsharp-Masking sichtbar positiven Einfluss auf die Blob-Detektierung. Diese Bildverarbeitungsoperationen verändern bekanntermaßen die Flankensteilheit von im Bild vorkommenden Objektkanten und somit den Bildkontrast.

5.3.4 Wasserzeichenextraktion aus Gray-Level-Blobs

Vor der Wasserzeichenextraktion findet eine Anpassung der Blob-Detektierung an Breite $\hat{X}$ und Höhe $\hat{Y}$ des Trägerbildes statt. Die Skalen ergeben sich zu $\bar{\sigma}_{\min} = \sigma_{\min} \cdot \max\left(\hat{X}, \hat{Y}\right) / 512$ und $\bar{\sigma}_{\max} = \sigma_{\max} \cdot \max\left(\hat{X}, \hat{Y}\right) / 512$.

Erneut werden nach einer *LOG*-Filterung des Bildes die optimalen Skalen $\hat{\sigma}_{opt}$ und die dazugehörigen Amplituden $\hat{A}_{opt}$ bestimmt. Wie bereits zuvor beschrieben, werden $K+1$ Blobs $\hat{B}_k := \left\{\hat{B}_k\left(x_k, y_k\right) : k = 0,1,...,K\right\}$ mit den größten Amplituden und den entsprechenden Mindestabständen zueinander ausgewählt (vgl. Abschnitt 5.3.2). Der größte Wert $\hat{B}_0$ stellt den Referenz-Blob dar, der für die Reihenfolge verantwortlich ist, mit der die Bits aus den K Daten-Blobs extrahiert werden.

Die Quantisierungsschrittweite $\hat{\Delta}$ wird auf Seiten der Extraktion nach Gleichung (5.12) berechnet. Die Kontrastnormierung findet dabei durch den Faktor $\hat{g}$ statt, welcher aus den Amplitudenwerten $\hat{A}_{opt}$ ermittelt wird. P ist der festgelegte (bekannte) Parameter für die Stärke der Wasserzeicheneinbettung.

$$\hat{g} = \left(\frac{1}{\hat{X} \cdot \hat{Y}} \sum_{x=1}^{\hat{X}} \sum_{y=1}^{\hat{Y}} \hat{A}_{opt}\left(x, y\right)^2 \right)^{1/2} \tag{5.11}$$

$$\hat{\Delta} = \hat{g} \cdot P / 100 \tag{5.12}$$

Die Wasserzeichenextraktion geschieht, wie in Abschnitt 2.1.1 beschrieben, durch eine Zuordnung der Trägersignalwerte zum jeweils nächstgelegenen Gitterpunkt des Gesamtquantisierungsgitters. Die Wasserzeichenbits $\hat{w}$ werden dabei durch Gleichung (2.3) aus dem Träger (Betragsamplituden der K ausgewählten Gray-Level-Blobs) ermittelt.

5.3.5 Robustheitsanalyse des Blob-basierten Wasserzeichenverfahrens

Um die Robustheit der insgesamt durch das Blob-basierte Wasserzeichenverfahren eingebetteten Daten zu bestimmen, wurden im Rahmen einer Simulation (siehe Anhang G.2) je Testbild $K+1 = 33$ Gray-Level-Blobs mit den größten *LOG*-Amplitudenwerten ausgewählt. Nach

einer Grundverstärkung G_{base} sowie einer Quantisierung der Blob-Amplituden (Einbetten der Datenbits) wurden die Abweichungen der jeweiligen *LOG*-Amplitudenwerte als Folge der Anwendung von Störungen berechnet.

Abbildung 5.13 zeigt einen Ausschnitt der Simulationsergebnisse aus Anhang G.2. Diese sind mit den in Abbildung 4.15 (S. 63) dargestellten Bitfehlerverhältnissen (BER) der in den Kapiteln 3 und 4 entwickelten adaptiven DWT-basierten Wasserzeicheneinbettung vergleichbar[60].

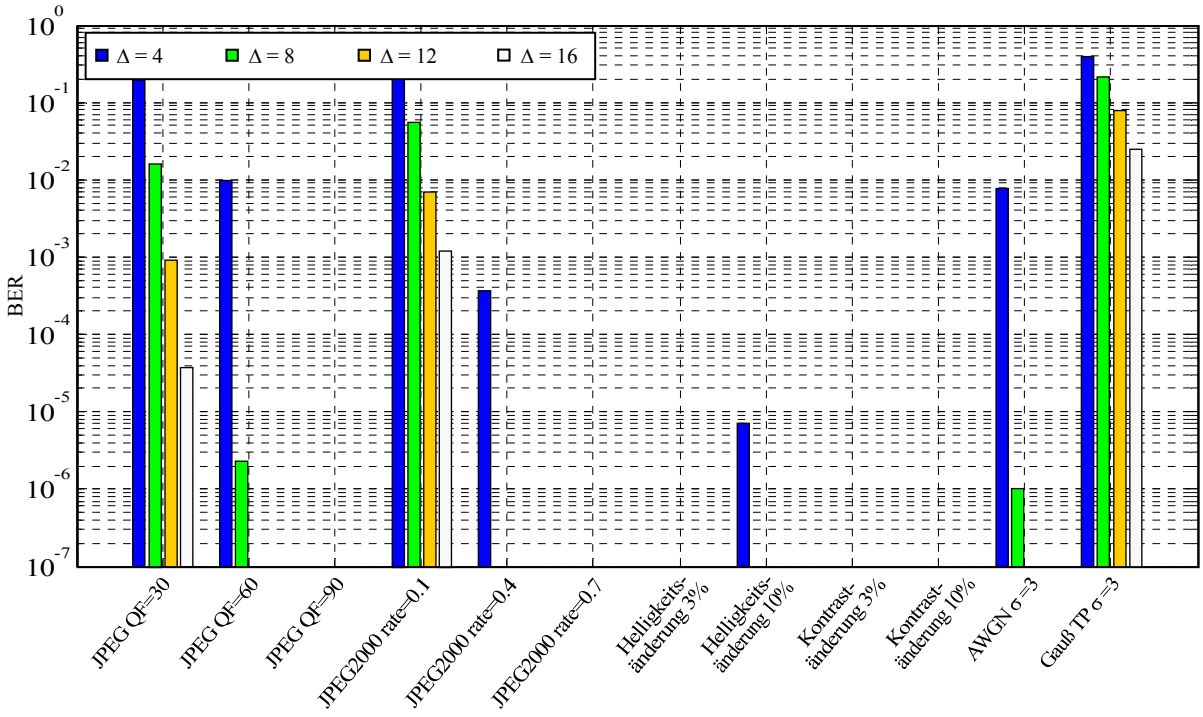


Abbildung 5.13 Robustheit des Blob-basierten Wasserzeichenverfahrens gegenüber Kompression, Helligkeits- und Kontraständerung, Gaußschem Rauschen und Tiefpassfilterung bei unterschiedlichen Einbettungsstärken Δ (Parameter: $G_{base} = 10$, $\sigma_{\min} = 3$, $\sigma_{\max} = 5$).

Anhand der ermittelten BER-Kurven ist zu erkennen, dass das Blob-basierte Wasserzeichenverfahren bereits eine hohe Robustheit gegenüber Kompression, Helligkeits- und Kontraständerungen sowie der Addition von Gaußschem Rauschen besitzt. Für geometrische Veränderungen des Trägerbildes, wie Rotation, Skalierung und Scherung, konnte die erwartete Robustheit jedoch nicht sofort erreicht werden.

Wie die Simulationsergebnisse im Anhang G.2 zeigen, kann es durch geometrische Trägerbildveränderungen zu einer starken Beeinflussung des Auswahlprozesses für die Gray-Level-Blobs kommen. Der Grund ist die im folgenden Abschnitt beschriebene Desynchronisation während der Extraktion der eingebetteten Wasserzeichendaten. Für eine Leistungssteigerung müssen diese Desynchronisationsfehler vermieden bzw. korrigiert werden.

[60] Die Einbettungsstärke wurde dazu auf die Quantisierungsschrittweite Δ und nicht den Parameter *P* bezogen.

5.4 Desynchronisation während der Blob-Auswahl

Wie bereits in Abschnitt 5.3.4 erwähnt, werden die Blobs auch bei der Wasserzeichenextraktion sukzessive ausgewählt. Zuerst wird der Blob $\hat{B}_0$ mit der größten *LOG*-Amplitude (Referenz-Blob) ermittelt. Anschließend werden die *K* Daten-Blobs mit jeweils nächst kleinerer Amplitude bestimmt. Dabei wird abermals um jeden ausgewählten Blob $\hat{B}_i$ ein Kreis der Größe $O \cdot \hat{\sigma}_{opt}\left(x_i, y_i\right)$ als reservierter Bereich markiert[61], aus dem kein anderer Blob ausgewählt werden kann. Das heißt, Ungleichung (5.5) muss für alle Blobs $\hat{B}_i$ und $\hat{B}_j$ der Liste ausgewählter Blobs erfüllt sein, wobei d_{ij} nach Gleichung (5.6) die Entfernung der beiden Blobs zueinander angibt.

5.4.1 Einfügung bzw. Auslöschung eines Blobs

Aufgrund von Störungen oder auch bereits durch Rundung oder Clipping der Grauwerte des markierten Bildes können die während der Einbettung ausgewählten Blobs ihre Position $\left(x, y\right)$ leicht verändern. Zwei Fälle, dargestellt in Abbildung 5.14, können dabei mit weit reichenden Folgen für den sukzessiven Blob-Auswahlprozess bei der Wasserzeichenextraktion auftreten:

1. Zwei Blobs, die sich beim Einbetten nicht überlappt haben (für die Gleichung (5.5) erfüllt war) können ihre Position geringfügig verändern, so dass $\hat{B}_j$ den bereits ausgewählten Blob $\hat{B}_i$ überlappt. $\hat{B}_j$ wird infolgedessen nicht ausgewählt, es kommt zu einer so genannten Auslöschung.
2. Zwei Blobs, die sich beim Einbetten überlappt hätten (für die Gleichung (5.5) nicht erfüllt war) können ihre Position leicht verändern, so dass $\hat{B}_j$ neben $\hat{B}_i$ auch ausgewählt wird. In $\hat{B}_j$ wurde allerdings kein Wasserzeichenbit eingebettet. Es handelt sich somit um eine Einfügung eines zufälligen Wertes.

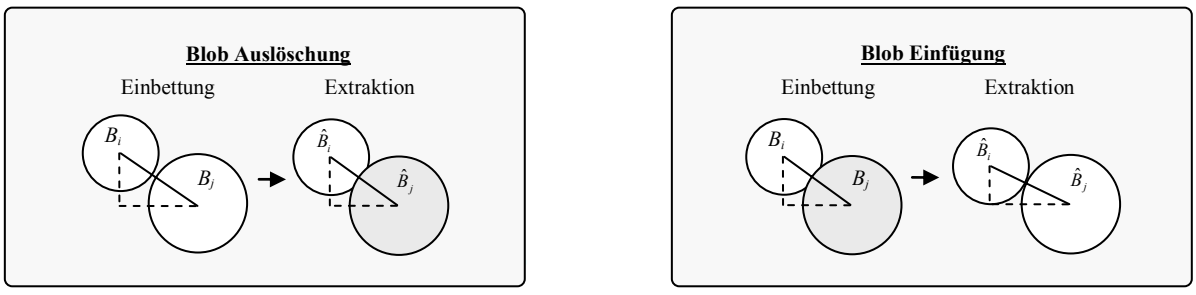


Abbildung 5.14 Blob-Auslöschung (links) bzw. Blob-Einfügung (rechts) während des sukzessiven Auswahlprozesses bei der Wasserzeichenextraktion

[61] Maskenmultiplikator $O = 6$ (vgl. S. 78).

Mögliche Konsequenzen einer derartigen Einfügung oder Auslöschung können sein,

- dass die Reihenfolge aller ab der Fehlstelle ausgewählten Blobs (extrahierten Wasserzeichenbits) um eins erhöht bzw. verringert wird, woraufhin eine Desynchronisation innerhalb der Wasserzeichensequenz auftritt oder
- dass der Auswahlprozess aller anschließend zu selektierenden Blobs gestört und damit die Reihenfolge an weiteren Stellen verändert wird.

Gewöhnliche Fehlerkorrekturverfahren können jedoch nur Substitutionsfehler (binär: 0→1 oder 1→0) korrigieren. Sie sind nicht in der Lage, Einfügungen oder Auslöschungen zu erkennen bzw. zu korrigieren.

5.4.2 Auswirkungen von Blob-Einfügungen und -Auslöschungen

Abbildung 5.15 veranschaulicht die Konsequenz einer Blob-Einfügung bzw. -Auslöschung. Die Reihenfolge aller anschließend ausgewählten Blobs ist um Eins erhöht bzw. verringert. Die Wasserzeichensequenz ist damit ab dieser Stelle nicht mehr synchron zur eingebetteten Sequenz. Demzufolge können die Wasserzeichenbits nicht mehr korrekt ausgelesen werden.

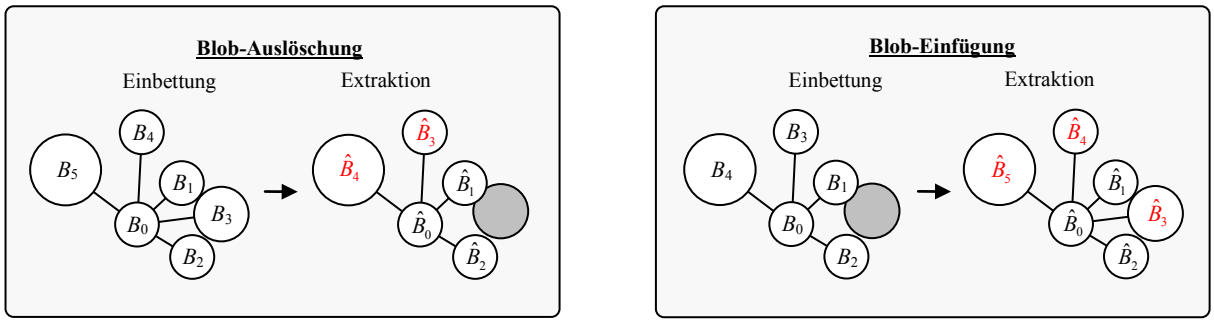


Abbildung 5.15 Konsequenz einer Blob-Auslöschung (links) bzw. Blob-Einfügung (rechts): Reihenfolge der anschließenden Blobs verringert bzw. erhöht sich

Liegen darüber hinaus mehrere Blobs im Bild sehr dicht beieinander und kommt es zu einer Einfügung/Auslöschung, dann können weitere Störungen der Blob-Auswahl auftreten. Wie exemplarisch in Abbildung 5.16 dargestellt, kann es vorkommen, dass der grau-markierte Blob, der beim Einbetten nicht ausgewählt wurde, bei der Extraktion Gleichung (5.5) erfüllt. Angenommen, der *LOG*-Amplitudenwert ist größer als der von Blob B_2, dann wird er an dessen Stelle ausgewählt. Im zweiten Szenario führt der eingefügte Blob zu weiteren Problemen. Aufgrund einer ähnlichen Entfernung zum Referenz-Blob $\hat{B}_0$, verglichen mit Blob $\hat{B}_3$, ergibt sich ein Wechsel innerhalb der Reihenfolge der Einbettungspositionen.

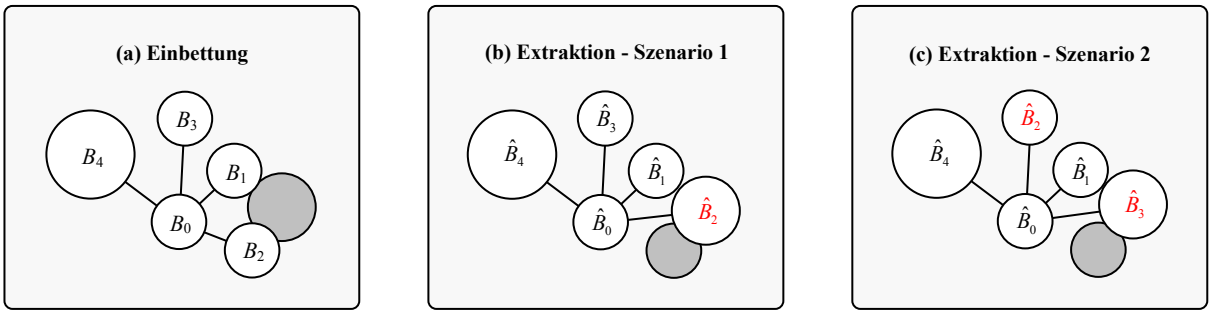


Abbildung 5.16 Bei der Einbettung ausgewählte Blobs (a), Auslöschung eines Blobs aufgrund einer Einfügung (b), Auslöschung und Vertauschung der Reihenfolge innerhalb der Blob-Liste aufgrund einer Einfügung (c)

Dieses grundsätzliche Problem der Desynchronisation tritt bei allen Wasserzeichenverfahren der zweiten Generation auf, die das Trägersignal nicht an einem festen Raster ausrichten. Bislang existiert keine allgemein zufrieden stellende Lösung. Motiviert dadurch wurde im Rahmen dieser Arbeit ein Verfahren zur Resynchronisation neu entwickelt, welches im Kapitel 6 vorgestellt wird. Die resultierende Leistungsfähigkeit des auf diese Weise erweiterten Blob-basierten Wasserzeichenverfahrens wird im Abschnitt 6.4 untersucht und mit den Daten ähnlicher Verfahren anderer Autoren verglichen.

5.5 Fazit

In den vorhergehenden Abschnitten wurde gezeigt, dass, basierend auf der Detektierung und gezielten Veränderung so genannter Gray-Level-Blobs, Wasserzeichendaten mit hoher Robustheit gegenüber einer Vielzahl unterschiedlicher Störungen in einem Bild eingebettet werden können.

Bei den verwendeten Gray-Level-Blobs handelt es sich um Texturelemente (Texel), die sich durch gute Texturmaskierungseigenschaften auszeichnen. Anhand von Beispielbildern wurde in diesem Kapitel demonstriert, dass eine gezielte Veränderung dieser Texturelemente zur Einbettung von Daten aufgrund der Texturmaskierung für das menschliche Auge nur schwer wahrnehmbar ist.

Im Zuge der Implementierung und Evaluierung des vorgestellten Blob-basierten Wasserzeichenverfahrens stellte sich heraus, dass es bei diesem speziellen, neuen Ansatz zu Desynchronisationsproblemen während der Wasserzeichenextraktion kommen kann. Aus diesem Grund wird im nachfolgenden Kapitel 6 die bereits in den Kapiteln 3 und 4 eingesetzte Fehlerkorrektur (Faltungscodierung mit Viterbi-Decodierung) erweitert. Dabei werden die Algorithmen eines Viterbi-Decoders zu Zwecken der Resynchronisation gezielt verändert.

Kapitel 6

Resynchronisation während der Gray-Level-Blob-Detektierung

In diesem Kapitel wird eine Resynchronisation von Einfügungen und Auslöschungen als Erweiterung des Gray-Level-Blob-basierten Wasserzeichenverfahrens neu entwickelt. Zunächst erfolgen eine Modifizierung des Blob-Auswahlprozesses und ein kurzer Überblick derzeit bekannter Ansätze zur Resynchronisation in selektiven Wasserzeichensystemen. Nachfolgend wird das entwickelte, auf dynamischer Programmierung beruhende Resynchronisationsverfahren im Detail erläutert, getestet und bezüglich der Leistungsfähigkeit mit den Ansätzen anderer Autoren verglichen. Abschließend wird die neue Resynchronisation mit dem in dieser Arbeit vorgestellten Gray-Level-Blob-basierten Wasserzeichenverfahren vereint und als Gesamtsystem ausführlich analysiert.

6.1 Erweiterung des Blob-Auswahlprozesses

6.1.1 Erlaubte Blob-Überlappung während der Wasserzeichenextraktion

Während des Wasserzeicheneinbettens mithilfe des in Kapitel 5 entwickelten Gray-Level-Blob-basierten Wasserzeichenverfahrens muss Gleichung (5.5) erfüllt sein. Jedoch können zwei ausgewählte Blobs, die während des Einbettens im Bild dicht zusammen liegen, einander auf Seiten der Extraktion aufgrund von Störungen überlappen (siehe Abschnitt 5.4).

Da es sich bei diesen dicht zusammen liegenden Blobs während der Wasserzeichenextraktion um mögliche Kandidaten für Einfügungen bzw. Auslöschungen handelt, wird der Blob-Auswahlprozesses aufgrund der gewonnenen Erkenntnisse wie folgt erweitert. Durch Gleichung (6.1) wird ein Überlappungsfaktor $C_{ij} := \left\{ C_{ij} \in \mathbb{R} : -\infty < C_{ij} < 1 \right\}$ als Erweiterung zu Gleichung (5.5) definiert, der während der Extraktion fortan auch Werte größer Null annehmen kann.

$$C_{ij} = \frac{\hat{\sigma}_{opt}\left(x_i, y_i\right) + \hat{\sigma}_{opt}\left(x_j, y_j\right) - d_{ij}}{\hat{\sigma}_{opt}\left(x_i, y_i\right) + \hat{\sigma}_{opt}\left(x_j, y_j\right)} \tag{6.1}$$

Der Überlappungsfaktor ist auf die Skalen der Blobs $\hat{B}_i$ und $\hat{B}_j$ normiert[62]. Ein Wert $C_{ij} = 0,01$ bedeutet z.B., dass die Blobs sich während der Extraktion zu 1% überlappen.

---

[62] $\hat{B}_i$ und $\hat{B}_j$ bezeichnen die *LOG*-Amplitudenwerte der i bereits ausgewählten Blobs und des durch den Auswahlprozess zu entscheidenden j-ten Blobs, wobei $j = i + 1$ ist (siehe Abschnitt 5.3.2).

6.1.2 Kandidaten für Einfügungen und Auslöschungen

Ist der Betrag $|C|$ für zwei benachbarte Blobs sehr klein, bedeutet dies eine hohe Wahrscheinlichkeit für das Auftreten einer Einfügung bzw. Auslöschung. Tendiert C im Gegensatz dazu im Bereich der negativen Zahlen betragsmäßig zu größeren Werten, so liegen die Blobs weit auseinander und weder eine Einfügung noch eine Auslöschung sind möglich. Genau entgegengesetzt verhält es sich, wenn der Überlappungsfaktor positiv ist und zwei Blobs sich sehr stark überlappen. Dann hat eine Auslöschung wahrscheinlich ebenfalls nicht stattgefunden und die betrachteten Blobs haben sich auch schon während des Einbettens überlappt.

Zusätzlich zum Referenz-Blob $\hat{B}_0$ werden daher sukzessive nicht nur K Blobs ausgewählt, sondern unter Einhaltung der im Folgenden aufgeführten drei Kriterien I - III insgesamt $K + D_{\mathrm{I}} + D_{\mathrm{E}}$ größte Amplitudenwerte der Matrix $\hat{A}_{opt}$. Dabei sind D_{I} Kandidaten für Einfügungen ($C_{ij} \le 0$) und D_{E} Kandidaten für Auslöschungen ($C_{ij} > 0$). Um diese zusätzlichen Kandidaten einzugrenzen, wird ein Überlappungsschwellwert $\tau \in \mathbb{R}^+$ definiert (bspw. $\tau = 0,01$).

I	**Liste fest ausgewählter Blobs** Sukzessive Auswahl von K Blobs, die zu den jeweils zuvor ausgewählten Blobs dieser Liste die Bedingung $C_{ij} \le -\tau : 0 \le j \le K$ erfüllen
II	**Liste der Kandidaten für Einfügungen D_{I}** Nach der Auswahl von K festen Blobs, sukzessive Auswahl all derjenigen Blobs, die zu den fest ausgewählten Blobs der ersten Liste die Bedingung $C_{ij} \le 0 : 0 \le j \le K + D_{\mathrm{I}}$ erfüllen
III	**Liste der Kandidaten für Auslöschungen D_{E}** Auswahl aller Blobs, die zu den fest ausgewählten Blobs der ersten Liste die Bedingung $0 < C_{ij} \le \tau : 0 \le j \le D_{\mathrm{E}}$ erfüllen

Tabelle 6.1 Kriterien des erweiterten Blob-Auswahlprozesses

Der Überlappungsfaktor C wird für alle ausgewählten Blobs und Kandidaten notiert[63] und anschließend bei der in dieser Arbeit entwickelten, resynchronisierenden Fehlerkorrektur zur Wichtung verwendet. Durch diesen Wert erhält die Fehlerkorrektur eine zusätzliche Information darüber, wie dicht die Blobs nebeneinander liegen und wie wahrscheinlich damit eine Einfügung oder Auslöschung für einen Blob-Kandidaten ist.

Abbildung 6.1 zeigt ein Beispielbild mit $K + 1 = 33$ ausgewählten Blobs während der Wasserzeicheneinbettung und nach einer Bildrotation. Ohne die beschriebene Erweiterung der Blob-Auswahl wird ein Blob aufgrund einer Auslöschung im linken oberen Bereich der mittleren Grafik zu wenig und ein Blob im rechten Bereich zu viel ausgewählt. Mit der Erweiterung des Blob-Auswahlprozesses werden an diesen Stellen des Bildes Kandidaten für Auslöschungen,

[63] Für jeden ausgewählten Blob wird nur der größte Überlappungsfaktor (d.h., zum dichtesten Nachbarn) notiert.

markiert durch das Symbol , sowie ein Kandidat für eine Einfügung, markiert durch das Symbol , ausgewählt.

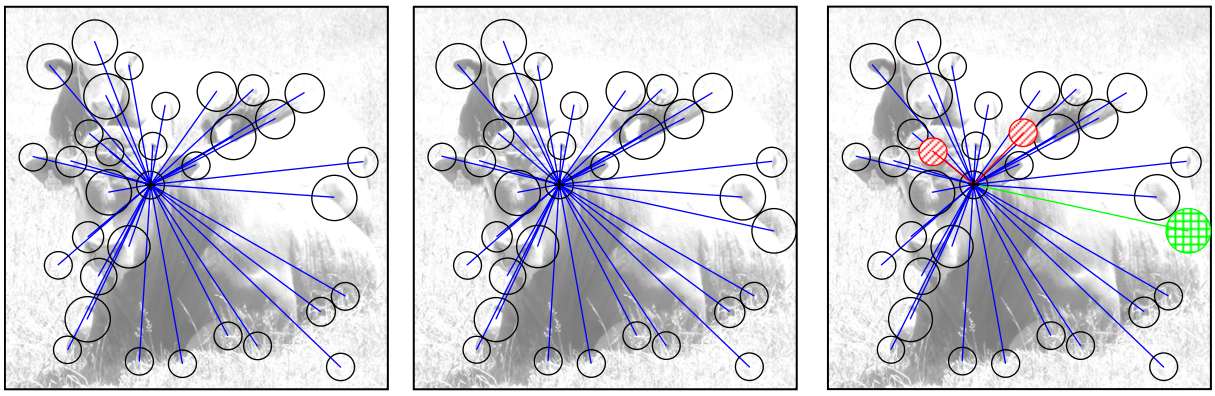


Abbildung 6.1 Beispielbild "tb01" mit $K+1=33$ ausgewählten Blobs während der Wasserzeicheneinbettung (links) und während der Wasserzeichenextraktion nach einer Rotation des Bildes um einen Winkel von 108° ohne Erweiterung der Blob-Auswahl (mittig) sowie mit Erweiterung der Blob-Auswahl (rechts)

Die Kandidaten für Einfügungen und Auslöschungen müssen nun von einer resynchronisierenden Fehlerkorrektur als richtige Daten-Blobs erkannt oder verworfen werden.

6.2 State-of-the-Art: Verfahren zur Resynchronisation

Bevor in Abschnitt 6.3 das im Rahmen dieser Arbeit neu entwickelte Korrekturverfahren beschrieben wird, gibt dieser Abschnitt einen Überblick bekannter Fehlerkorrekturverfahren, die in der Lage sind, neben gewöhnlichen Bitfehlern auch Einfügungen und Auslöschungen zu korrigieren. Diese Verfahren lassen sich in drei Kategorien unterteilen[64]: verkettete und punktierte Fehlerkorrekturcodes sowie Techniken der dynamischen Programmierung.

6.2.1 Codeverkettung zur Korrektur von IDS-Fehlern

Eine Möglichkeit der Korrektur von IDS-Fehlern[65] bietet die Anwendung *verketteter Codes* [For66], vorrangig eingesetzt bei der Übertragung von Satellitendaten. Dabei wird die Nachricht (einzubettende Daten) zuerst mithilfe eines äußeren Codes und dann durch einen inneren Code codiert (siehe Abbildung 6.2). Als äußere Codes werden Fehlerkorrekturtechniken eingesetzt, die gut mit bündelartigen Fehlern zurechtkommen, wie bspw. *Reed-Solomon-Codes*[66] oder *Low-Density-Parity-Check-Codes*. Der innere Code ist ein Faltungscode, der Einzelfehler gut korrigieren kann, bei Bündelfehlern oder abschnittsweiser Desynchronisation jedoch

[64] vgl. [SPM08]

[65] Die Begriffe Einfügung, Auslöschung und gewöhnlicher 0→1 bzw. 1→0 Bitfehler werden im Folgenden mit IDS-Fehler abgekürzt (engl. *insertion, deletion, substitution*).

[66] Abk. *RS-Codes* (siehe [Bos98])

versagt. Diese Bündelfehler werden bei verketteten Codes durch den nachgeschalteten äußeren Code abgefangen, wodurch beide Techniken ihre Eigenschaften vereinen (siehe [Bos98]).

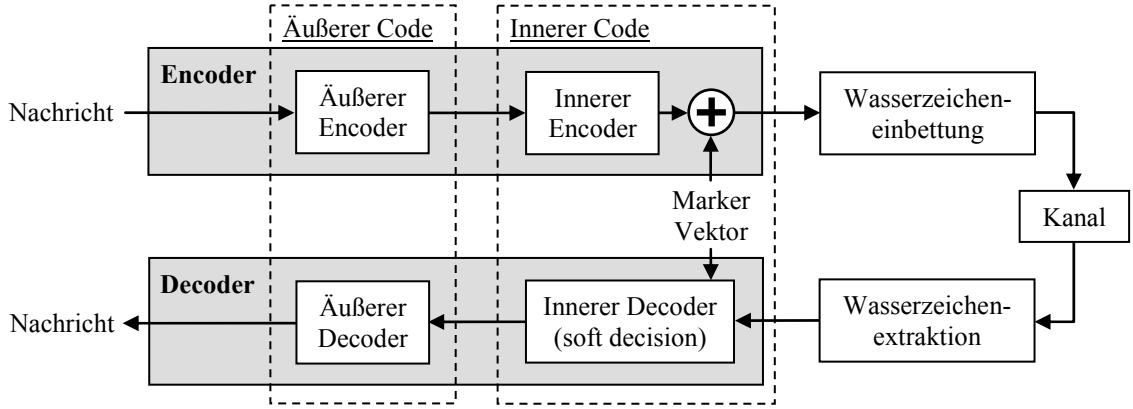


Abbildung 6.2 Codeverkettung eines inneren und äußeren Codes zur Korrektur von IDS-Fehlern

In [DM01] beschreiben Davey und MacKay erstmals eine Verkettung von Codes zur Korrektur von Einfügungen und Auslöschungen. Sie platzieren ein *Markersignal* im inneren Codestrom und unterteilen diesen somit in detektierbare Pakete fester Länge. Wird eine Desynchronisation entdeckt, kann innerhalb des Paketes nach den exakten Fehlstellen gesucht werden. Aufgegriffen wird dieser Ansatz später u.a. von Sharma und Coumou in [CS08] zum Einsatz in einem Wasserzeichensystem mit selektiver Einbettung.

6.2.2 Punktierung zur Korrektur von IDS-Fehlern

Ein anderer Ansatz zur Korrektur von IDS-Fehlern ist die *Punktierte Codierung*[67], bekannt aus der Kanalcodierung [Bos98]. Dabei werden bestimmte Symbolpositionen eines Codewortes ausgeblendet (nicht übertragen). Sie werden dem Decoder bekannt gemacht, so dass er vor dem Decodierprozess "don't care"-Werte[68] in das Codewort einsetzen kann. Auf diese Weise können die Codewortlängen flexibel an eine bestimmte Rahmenlänge bzw. Coderate angepasst werden. Durch die Punktierung im Verhältnis[69] K / N wird zwar die Leistungsfähigkeit des Originalcodes der Rate R_c verringert, der punktierte Code ist jedoch i. Allg. genauso gut wie unpunktierte Codes der Rate $(N-K) \cdot R_c$.

Diese Technik lässt sich auch zur Korrektur von IDS-Fehlern in einem Wasserzeichensystem mit selektiver Einbettung nutzen. So stellen Solanki *et. al*. in [SJM+04] zwei Ansätze vor, die das Trägersignal an Stellen, an denen keine Daten eingebettet sind, als punktierte Stellen

[67] engl. *punctured channel coding*

[68] Bei bipolaren Codesymbolen $[-1,+1]$ bildet der Wert 0 den "don't care"-Zustand.

[69] N ist die Anzahl aller Trägersignalpositionen. K ist die Anzahl der für das Einbetten ausgewählten Positionen.

behandeln. Sie unterteilen ein Bild in 8x8-Pixelblöcke und betten nur dort Daten ein, wo ein aus den DCT-Koeffizienten der 8x8-Blöcke berechnetes Energiemaß oder die Koeffizienten selbst einen festgelegten Schwellwert überschreiten. Die einzubettende Nachricht wird mit einem Reed-Solomon- oder einem *Repeat-Accumulate-Code*[70] niedriger Rate codiert. Die Blöcke bzw. Positionen, in denen aufgrund des Schwellwertkriteriums keine Bits eingebettet werden, bezeichnen die Autoren als *Erasures*. Auf Seiten der Extraktion wird dasselbe Schwellwertkriterium angewendet. Die Positionen, an denen der Detektor aufgrund des Kriteriums keine eingebetteten Daten erwartet, werden als "don't care"-Werte behandelt.

Abbildung 6.3 zeigt als Beispiel eine mithilfe der in Abschnitt 4.2.1 entwickelten Gradientenbasierten Bildsegmentierung berechnete Maske zum selektiven Einbetten von Daten[71]. Wird die Maske nach einer JPEG-Kompression des Bildes (*quality factor* QF = 50) erneut berechnet, ergeben sich für das betrachtete Bild neun Stellen, an denen sich die Maske ändert. Eine Auslöschung *e* führt zu einem "don't care"-Wert, und aus einer Einfügung wird ein Fehler *f*. Im gezeigten Beispielbild (Abbildung 6.3 (links)) kommt es somit zu sieben Einfügungen und zwei Auslöschungen, wenn der Decoder davon ausgeht, dass in den weißen Regionen Daten eingebettet sind.

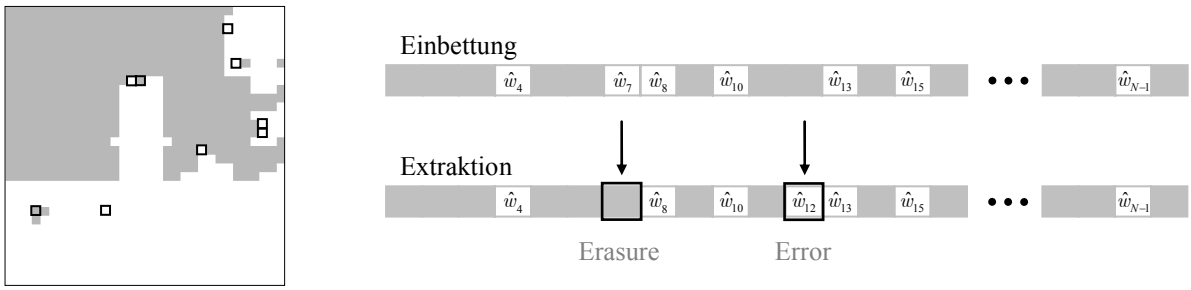


Abbildung 6.3 IDS-Korrektur durch Erasures auf Seiten der Wasserzeichenextraktion

Es ist bekannt, dass (η, κ)-RS-Codes der Rate $R_c = \kappa / \eta$ bis zu $e + 2f \leq \eta - \kappa$ Auslöschungen und Fehler korrigieren können (obere Grenze). In einem System mit punktierter Codierung stellen alle Positionen, an denen nicht eingebettet wird, eine Auslöschung dar. Somit muss in diesen Systemen stets eine Grundanzahl von $\tilde{e} = N - K$ Auslöschungen vom Decoder behandelt werden, wodurch sich die verbleibende Korrekturleistung nach Gleichung (6.2) ergibt.

$$N - K + e + 2f \leq \eta - \kappa \tag{6.2}$$

[70] Abk. *RA-Code* (siehe [Bos98])

[71] Im Rahmen dieser Arbeit wurde ein weiteres selektives Wasserzeichenverfahren entwickelt [SPM07b]. Analog zu dem in Kapitel 4 beschriebenen Ansatz, zwei unterschiedliche Quantisierungsschrittweiten zu verwenden, wurden die Daten ausschließlich in den stärker texturierten Bildbereichen eingebettet.

Im Falle der RA-Codierung lässt sich eine derartige Abschätzung analytisch nur mit großem Aufwand vornehmen. Im Rahmen dieser Arbeit wurde jedoch $5 \cdot R_c \leq K / N$ empirisch als geeignete obere Grenze ermittelt.

Für die RS-Codierung ist zu beachten, dass eine einzelne Auslöschung oder ein Einzelbitfehler stets alle Bits eines Codesymbols betrifft. Die Fehlerkorrekturleistung des punktierten RS-Codes nach Gleichung (6.2) sinkt daher drastisch, wenn nur ein Bit je Trägersignalposition eingebettet wird und ein Codesymbol sich über mehrere unabhängige Positionen erstreckt. Aus diesem Grund und auch wegen der höheren Einzelfehler-Korrekturleistung (unterstützt durch *soft-decision-decoding*) sind RA-Codes den RS-Codes vorzuziehen.

6.2.3 Dynamische Programmierung zur Korrektur von IDS-Fehlern

Noch ein paar Jahre vor Solanki *et al*. stellen Mansour und Tewfik in [MT01] einen weiteren Ansatz zur Korrektur von IDS-Fehlern in einem Wasserzeichensystem mit selektiver Einbettung vor. Ihre Technik beruht auf *Dynamischer Programmierung*. Sie ist in der Lage, die korrekte Nachricht auch dann wiederherzustellen, wenn zusätzliche Bits an tatsächlich unbekannten, zufälligen Stellen innerhalb der Wasserzeichensequenz auftreten. Mit "tatsächlich unbekannten Stellen" ist gemeint, dass das Decodiersystem nicht als Block-, sondern als Faltungsdecoder arbeitet. Die Gesamtlänge der Trägersequenz muss im Gegensatz zu den Verfahren von Solanki *et al*. oder Davey und MacKay am Decoder nicht bekannt sein.

Die Autoren modifizieren einen zur Decodierung von Faltungscodes verwendeten *Viterbi-Algorithmus* (siehe Abschnitt 2.3.2) derart, dass er zusätzlich zu gewöhnlichen Substitutionsfehlern auch Einfügungen korrigieren kann. Dazu erweitern sie den Viterbi-Decoder, wie in Abbildung 6.4 dargestellt, um ein paar Zustände und Pfade (vgl. Abbildung 2.11). Die Zustände Ⓘ, Ⓘ, Ⓘ, Ⓘ und die dickeren, schwarzen Pfade beschreiben den Original-Decoder. Hinzu kommen die Zustände Ⓘ̃, Ⓘ̃, Ⓘ̃, Ⓘ̃, die über die dünneren (roten und blauen) Pfade erreicht werden. Diese Pfade bedeuten, dass der modifizierte Viterbi-Decoder jeweils eine Einfügung für das erste bzw. zweite Symbol des Eingangswerte-Tupels annimmt und die Metriken auch für diese Fälle berechnet. Ist die Metrik nach einer dieser Abzweigungen den Metriken der dickeren, schwarzen Pfade zu den Zuständen Ⓘ, Ⓘ, Ⓘ, Ⓘ überlegen, dann wird das System des Original-Decoders verlassen, um einen der asynchronen Parallelzustände einzunehmen. Aus diesen zeitlich um eine Symbolposition asynchronen Parallelzuständen gelangt der Decoder direkt im Anschluss wieder zurück in die Zustände Ⓘ, Ⓘ, Ⓘ, Ⓘ, wobei der erworbene Versatz (Resynchronisation) beibehalten wird.

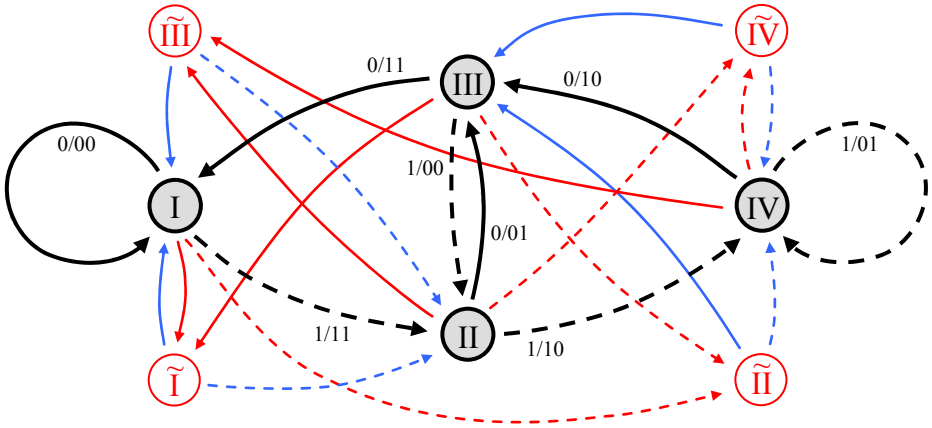


Abbildung 6.4 Erweitertes Zustandsdiagramm des von Mansour und Tewfik modifizierten Viterbi-Decoders zur Korrektur von Substitutionsfehlern und Einfügungen (vgl. Abbildung 2.11)

Der modifizierte Decoder ist allerdings nicht in der Lage, Symbol-Auslöschungen zu erkennen, was dazu zwingt, diese entsprechend zu vermeiden. So existiert in einem selektiven Wasserzeichensystem stets ein Kriterium, das bestimmt, ob ein Koeffizient, eine Region oder ein anderes Merkmal zum Einbetten ausgewählt oder ob es unverändert gelassen wird. Für das in dieser Arbeit entwickelte Blob-basierte Wasserzeichenverfahren entspricht das Auswahlkriterium dem Abstand des in Gleichung (6.1) definierten Überlappungsfaktors C_{ij} zum Entscheidungsschwellwert τ. Um in einem derartigen System, in dem sowohl Einfügungen als auch Auslöschungen gleichwahrscheinlich auftreten, mit höherer Wahrscheinlichkeit nur noch entweder das eine oder das andere zu erhalten, kann der Entscheidungsschwellwert auf Seiten der Extraktion, wie in Abbildung 6.5 gezeigt, verändert werden. Wenn d_τ der Abstand des Überlappungsfaktors vom Schwellwert ist und der Schwellwert während der Extraktion um den Wert T verringert wird, so verringert sich auch die Wahrscheinlichkeit für das Auftreten von Auslöschungen. Jedoch werden bei diesem Ansatz alle Trägersignalpositionen, deren Überlappungsfaktor während des Einbettens im Bereich $[-T;0]$ liegt, zu Einfügungen, wodurch der FEC-Decoder belastet wird.

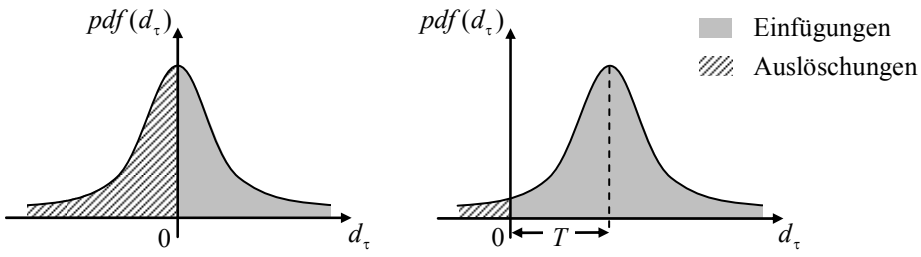


Abbildung 6.5 Verschiebung des Schwellwertes auf Seite der Extraktion zur Vermeidung von Auslöschungen

In [SFS04] stellen Swart *et al.* das Zusammenschalten mehrerer Viterbi-Decoder zu einem einzigen Decoder vor, den sie als Super-Trellis-Decoder bezeichnen. Dieser Decoder kann nun sowohl Einfügungen als auch Auslöschungen behandeln. Damit entfällt die Verschiebung

des Schwellwertes. Das Funktionsprinzip ist ähnlich wie bei dem von Mansour und Tewfik beschriebenen modifizierten Einzeldecoder, jedoch sind hier mehrere Viterbi-Decoder parallel geschaltet und untereinander verbunden.

Die drei separaten Viterbi-Decoder des Super-Trellis-Systems erhalten, wie in Abbildung 6.6 gezeigt, als *soft-input* jeweils die Symbolfolge $\hat{w} := \hat{w}_1, \hat{w}_2, ..., \hat{w}_K$. Dabei ist einer der Decoder dem "0"-Decoder zeitlich um ein Symbol im Voraus, dargestellt durch "+1". Ein zweiter Decoder liegt zeitlich um ein Symbol zurück, dargestellt durch "–1". Zu jedem Decodierzeitpunkt werden die Metriken aller drei Decoder berechnet (siehe Abschnitt 2.3.2). Derjenige Decoder, dessen Metrik den anderen überlegen ist und dadurch den wahrscheinlichsten Pfad anzeigt, wird zum "0"-Decoder, die anderen beiden Decoder werden aktualisiert. Bei einer erkannten Einfügung wird das Eingangssymbol verworfen. Bei einer erkannten Auslöschung wird das vorhergehende Symbol dupliziert und in die Folge eingefügt.

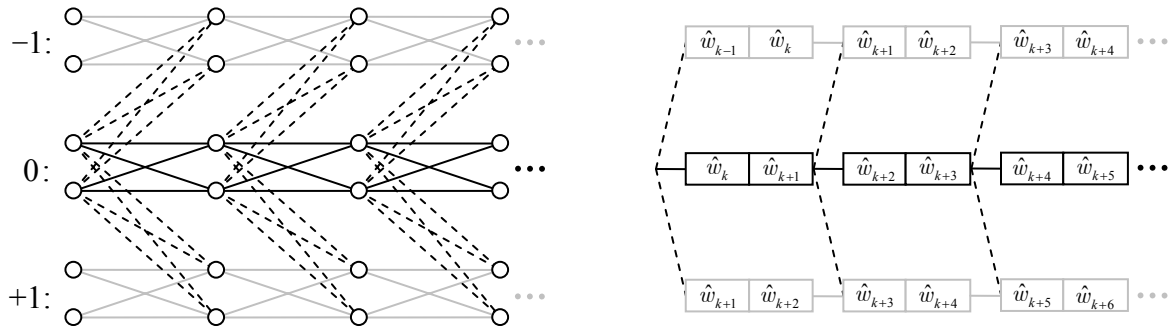


Abbildung 6.6 Abstrahierte Darstellung des Super-Trellis von Swart *et al*. [SFS04] (links) und andere Darstellung zur Demonstration der Auswahl der Eingangswerte für einen Code der Rate R_c = 1/2 (rechts)

6.3 Der modifizierte Super-Trellis-Decoder

Grundsätzlich stehen zur Resynchronisation einer Datenfolge die in Abschnitt 6.2 beschriebenen drei Ansätze zur Verfügung. Die punktierte Codierung von Solanki *et al*. [SJM+04] ist jedoch im Gegensatz zu den übrigen zwei Ansätzen an das Verhältnis[72] K / N gebunden. Es ist nur dann einsetzbar, wenn dieses Verhältnis nicht zu groß wird. Die empirisch bestimmte Coderate für die Korrektur von IDS-Einzelfehlern ist bei der punktierten Codierung mit ca. $5 \cdot R_c \leq K / N$ (RA-Coding) bzw. $N - K + e + 2f \leq \eta - \kappa$ (RS-Coding) vorgegeben. Da bei dem im Abschnitt 5.3 entwickelten Blob-basierten Wasserzeichenverfahren $K \ll N$ ist[73], kann die punktierte Codierung nicht verwendet werden.

[72] N ist die Anzahl aller Trägersignalpositionen. K ist die Anzahl der für das Einbetten ausgewählten Positionen.

[73] Bspw. für ein Bild der Größe 512x512 Pixel ist N = 262144. Die K = 32 Blobs können gleichwahrscheinlich an allen Positionen der ebenfalls 512x512 Werte großen Matrix $\hat{A}_{opt}$ ausgewählt werden.

Der von Davey und MacKay in [DM01] beschriebene Ansatz der verketteten Codierung ist ebenfalls nicht für das Blob-basierte Wasserzeichenverfahren geeignet. Das Verfahren arbeitet mit Blocklängen von mehreren tausend Bits. Die Leistungsfähigkeit ist bei kleinen Blocklängen (bspw. $K = 32$) stark eingeschränkt. In ihrer Veröffentlichung schreiben die Autoren, dass der Einsatz eines Viterbi-Algorithmus im Sinne der dynamischen Programmierung zur Korrektur von IDS-Fehlern bei kurzen Nachrichtenlängen deutlich besser geeignet ist.

Der von Swart *et al*. in [SFS04] vorgestellte Super-Trellis-Decoder auf Basis der dynamischen Programmierung ist nicht an das Verhältnis K / N gebunden. Die Leistungsfähigkeit ist nicht durch die Länge der Nachricht K eingeschränkt. Er kann somit prinzipiell für das Blob-basierte Wasserzeichenverfahren eingesetzt werden.

6.3.1 Veränderungen des Super-Trellis-Decoders

Der von Swart *et al*. beschriebene Super-Trellis-Decoder wurde im Rahmen dieser Arbeit in sechs Punkten verändert, wodurch die Leistungsfähigkeit um ein Vielfaches gesteigert werden konnte. Er erhält für die ersten vier, bereits in [SPM07b] vorgestellten Punkte im Folgenden die Bezeichnung MSTrellis1. Die Erweiterungen um die Punkte fünf und sechs, beschrieben in [SPM08], führen zu den Bezeichnungen MSTrellis2 und MSTrellis3. Ein Vergleich dieser drei neuen Verfahren (implementiert für eine Coderate $R_c = 1/2$) mit den Verfahren der anderen Autoren befindet sich im Abschnitt 6.3.2.

1. Es seien $p := \left\{ p_k \in \mathbb{N}^+ : C(p_k) \le 0,\ 1 \le p_k \le N, 1 \le k \le K \right\}$ die Positionen, an denen im Trägersignal $x := \left\{ x_n \in \mathbb{R} : 1 \le n \le N \right\}$ aufgrund des Selektionskriteriums $C(p_k)$ Wasserzeichenbits $w := w_1, w_2, ..., w_K$ eingebettet sind. Die Positionen, an denen nichts eingebettet ist, sind demzufolge $\overline{p} := \left\{ \overline{p}_k \in \mathbb{N}^+ \setminus p_k : 1 \le \overline{p}_k \le N, 1 \le k \le N - K \right\}$.

   Wenn der Decoder von Swart *et al*. im Bereich $(p_k ; p_{k+1})$ der Trägersignalpositionen zu der Annahme gelangt, dass eine Auslöschung stattgefunden hat, dann wird das Symbol $\hat{w}_{k+1}$ dupliziert eingefügt (vgl. Abbildung 6.6). Auf der Decoderseite steht jedoch das Wissen über alle Trägersignalwerte zur Verfügung. Es ist also weitaus sinnvoller, einen Wert $e = Q\left(\hat{A}_{opt}(\overline{p})\right)$ an der Position der Auslöschung $\overline{p}$ zu verwenden, der tatsächlich durch den Quantisierer $Q(\bullet)$ geliefert wurde, wobei $p_k < \overline{p} < p_{k+1}$. Kommen mehrere Trägersignalwerte infrage $(p_k < p_{k+1} - n : 2 \le n)$, kann diejenige Position $\overline{p}_n$ ausgewählt werden, deren Selektionskriterium den geringsten Abstand zur Entscheidungsschwelle hat.

$$e = Q\left(\hat{A}_{opt}\left(p_k + \arg\min_n\left(C(\overline{p}_n)\right)\right)\right) \tag{6.3}$$

2. Zu einer Auslöschung kann es zwischen den beiden Trägersignalpositionen p_k und p_{k+1} nur dann gekommen sein, wenn $p_k < p_{k+1} - 1$ ist. Somit reicht es aus, nur diese Stellen

vom Algorithmus untersuchen zu lassen. Die Folgen sind eine Reduzierung des Rechenaufwandes und der Anzahl möglicher Fehlentscheidungen des Decoders.

Ebenfalls werden der Rechenaufwand und die Fehleranfälligkeit reduziert, indem der Decoder nur an den Stellen eine Resynchronisation berechnet, die als Kandidaten für Einfügungen/Auslöschungen infrage kommen (siehe erweiterte Auswahlkriterien I - III, Abschnitt 6.1)

3. Der ursprüngliche Super-Trellis-Decoder ist nicht in der Lage, die tatsächliche Position einer Einfügung/Auslöschung innerhalb des Eingangswerte-Tupels $\left[\hat{w}_k, \hat{w}_{k+1}\right]$ zu bestimmen. Der in dieser Arbeit neu entwickelte, modifizierte Decoder hingegen rechnet, wie in Abbildung 6.7 angedeutet, alle möglichen Kombinationen durch und ermittelt exakt die tatsächliche Fehlerposition. Um den Rechenaufwand für dieses "Ausprobieren" zu begrenzen, erfolgt die Eingrenzung durch ein Betrachtungsfenster einer festgelegten Fensterlänge[74]. Während der Simulationen wurde die Fensterlänge = 5 gewählt. Der Decoder von Swart *et al*. hingegen trifft alle Entscheidungen bereits (sehr früh) nach einer Fensterlänge = 1. Dadurch ist der Rechenaufwand klein, aber die Wahrscheinlichkeit für Fehlentscheidungen ist entsprechend hoch.

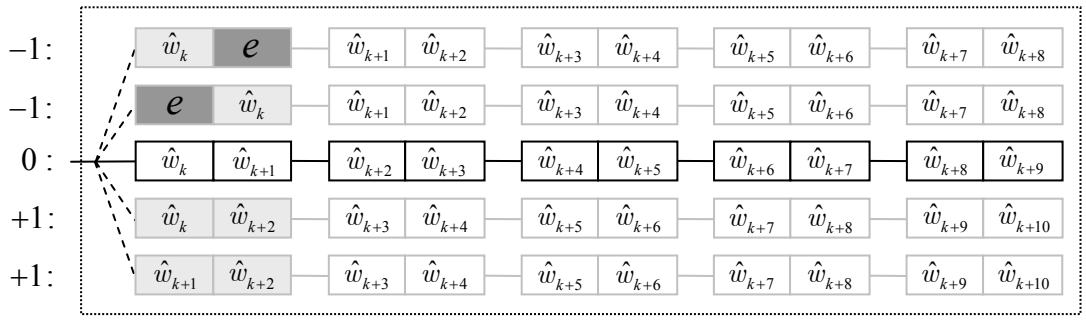


Abbildung 6.7 MSTrellis1-Decoder (R_c = 1/2, Fensterlänge = 5), wobei $e = Q\left(\hat{A}_{opt}\left(\overline{p}\right)\right)$

4. Durch die im Abschnitt 4.2.3 entwickelte Wichtung[75] des Decodereingangssignals (siehe Gleichung (6.4)) werden die Metriken der zeitlich asynchronen Trellis-Kombinationen beeinflusst.

$$\tilde{w} = \hat{w} \cdot f_1\left(C\right) \tag{6.4}$$

Dabei wird die Sicherheit der Entscheidung in Hinblick auf die Entfernung des Überlappungsfaktors C_{ij} zum Entscheidungsschwellwert τ berücksichtigt.

[74] Die Fensterlänge zur Entscheidungsfindung, ob eine Einfügung / Auslöschung stattgefunden hat, ist unabhängig von der Fensterlänge für den Vorwärts- / Rückwärtsschritt des Viterbi-Decoders (vgl. Abschnitt 2.3.2).

[75] Die Wichtungsfunktion $f_1\left(C\right)$ ergibt sich nach Gleichung (4.3).

5. In der durch die Punkte 1 bis 4 beschriebenen Variante des modifizierten Super-Trellis-Decoders MSTrellis1 kann innerhalb der Fensterlänge nur eine Einfügung / Auslöschung erkannt und korrigiert werden (siehe Abbildung 6.8).

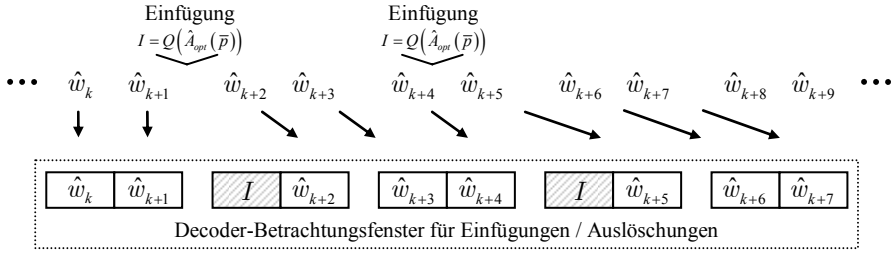


Abbildung 6.8 Beispiel zweier Einfügungen innerhalb der Fensterlänge 5

Um diese Einschränkung aufzuheben, wurde der Vorwärts-Algorithmus des Decoders, wie in [SPM08] beschrieben, zu einer rekursiven Baumstruktur weiterentwickelt. Dabei verändert sich die Darstellung des MSTrellis1 aus Abbildung 6.7 (um 90° gedreht) zu einer hierarchischen Baumstruktur, welche der Algorithmus in jedem Decodierschritt "Ast für Ast" durchläuft. In der dazugehörigen Abbildung 6.9 sind die Eingangswerte-Tupel für einige der Äste angegeben. Der jeweils linke Ast eines Knotens entspricht dabei dem Eingangswerte-Tupel ohne Einfügung / Auslöschung. Die anderen Äste stehen für die möglichen Kombinationen von Einfügungen oder Auslöschungen, die zu einem zeitlichen Versatz der nachfolgenden Eingangssymbole führen.

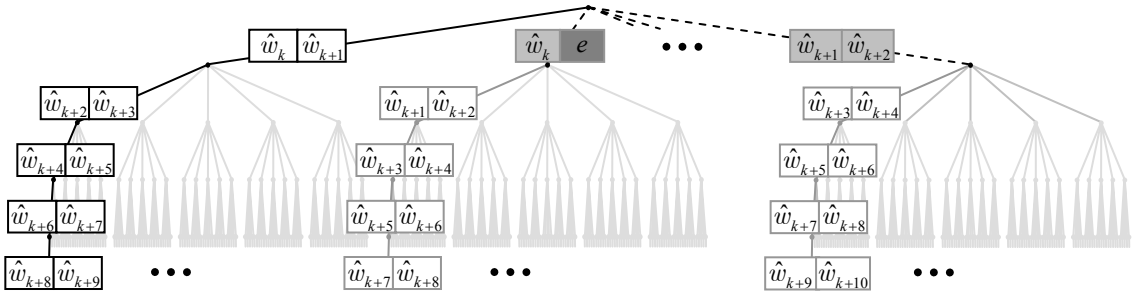


Abbildung 6.9 Baumstruktur des modifizierten Super-Trellis-Decoders MSTrellis2

Vervollständigt man die Eingangswerte-Tupel der Baumstruktur, so fällt auf, dass sich viele Äste gleichen. Auch, wenn das Betrachtungsfenster nach der erfolgten Synchronisationsentscheidung im anschließenden Decodierschritt das nächste Symbolpaar aufnimmt, ergeben sich für sehr viele Äste der Baumstruktur die gleichen Berechnungsschritte mit den gleichen Eingangswerten wie im vorherigen Decodierschritt.

Bei der Entwicklung und der in der Programmierumgebung *MATLAB* zu Simulationszwecken getätigten Implementierung des vorgestellten Verfahrens MSTrellis2 wurden diese nach jedem Decodierschritt gleichen Berechnungen berücksichtigt. Die zu jedem

Decodierzeitpunkt berechneten Metriken (siehe Abschnitt 2.3.2) ergeben für gleiche Äste dieselben Werte. Die Berechnung muss entsprechend nur einmal durchgeführt werden, wodurch sich eine erhebliche Rechenzeitersparnis ergibt.

6. Es seien p_I die Positionen der Einfügungen und p_E die Positionen der Auslöschungen, die der MSTrellis2-Decoder nach der Abarbeitung der Eingangssymbolfolge bestimmt hat. Indem die Werte $C\left(p_I \cup p_E\right)$ an diesen Positionen invertiert werden, ergibt sich eine korrigierte Auswahl der Trägersignalpositionen $\tilde{p} = \left(p \cup p_E\right) \setminus p_I$. Diese aktualisierte Auswahl $\hat{w} = Q\left(\hat{A}_{opt}\left(\tilde{p}\right)\right)$ kann dem MSTrellis2-Decoder ein zweites Mal zugeführt werden. Simulationen haben gezeigt, dass dadurch eine nochmalige Leistungssteigerung erreicht werden kann (siehe Anhang H.1). Dieses Verfahren mit zwei Korrekturdurchläufen wird im Folgenden als MSTrellis3-Decoder bezeichnet.

6.3.2 Leistungsfähigkeit des modifizierten Super-Trellis-Decoders

Die Leistungsfähigkeit des neu entwickelten Verfahrens zur Korrektur von Einfügungen und Auslöschungen wurde anhand umfangreicher Simulationen untersucht. Dabei wurde die Resynchronisation zunächst einzeln betrachtet (ohne Wasserzeicheneinbettung).

Um die Fehlerkorrekturleistung mit den Daten der Verfahren anderer Autoren vergleichen zu können, muss dasselbe Kanalmodell (*binary symmetric channel* (BSC)) verwendet werden wie in [DM01], [SFS04] und [CS08]. Die Auftretenswahrscheinlichkeit für Substitutionsfehler P_f wird dabei durch das aus der Kanalcodierung bekannte Bitenergie-Rauschleistungsdichte-Verhältnis[76] E_b/N_0 angegeben (Gleichung (6.5)). Auf diese Weise wird auch bei unterschiedlichen Modulationsverfahren ein Vergleich der Leistungsdaten ermöglicht.

$$P_f = 0{,}5 \cdot \operatorname{erfc}\left(\sqrt{\frac{E_b}{N_0}}\right) \tag{6.5}$$

Geht das Verhältnis E_b/N_0 gegen Unendlich, dann geht die Auftretenswahrscheinlichkeit für Substitutionsfehler P_f gegen Null.

Substitutionsfehler werden in der Beschreibung durch dieses Kanalmodell als eine Überlagerung des ungestörten Wasserzeichensignals w mit weißem Gaußschen Rauschen dargestellt, dessen Mittelwert Null ist und dessen Varianz σ^2 durch Gleichung (6.6) beschrieben wird.

$$\sigma^2 = 10^{-\left(\frac{E_b/N_0}{10}\right)} \tag{6.6}$$

Das durch das Rauschsignal z gestörte Wasserzeichensignal ist $\hat{w} = w + z$.

[76] engl. *energy per bit to noise power spectral density ratio*

Zur Simulation von Desynchronisationsfehlern wird eine gemeinsame Wahrscheinlichkeit für das Auftreten von Einfügungen und Auslöschungen P_d definiert. Ist die Länge des eingebetteten Wasserzeichens bspw. $K = 1000$ und $P_d = 1\%$, dann werden während der Simulation an 10 zufällig ausgewählten Positionen innerhalb der Sequenz $\hat{w} := \hat{w}_1, \hat{w}_2, ..., \hat{w}_K$ Symbole entfernt bzw. eingefügt.

Für die Simulation werden neben der Sequenz $\hat{w}$ auch Werte C im Bereich $(-\infty; +1)$ für die Überlappung benötigt. Diese werden je Symbol $\hat{w}_k$ als Zufallswerte generiert, wobei im Falle einer Auslöschung $0 < C_k \leq \tau$ und im Falle einer Einfügung $-\tau < C_k \leq 0$ eingesetzt wird.

Die Ergebnisse der Simulation befinden sich im Anhang H.1. Eine Zusammenfassung ist in Abbildung 6.10 dargestellt. Ebenfalls sind in dieser Darstellung die in [SFS04] und [DM01] angegebenen Simulationsergebnisse eingetragen. Die Auftretenswahrscheinlichkeit für Substitutionsfehler P_f ist in diesem Leistungsvergleich Null.

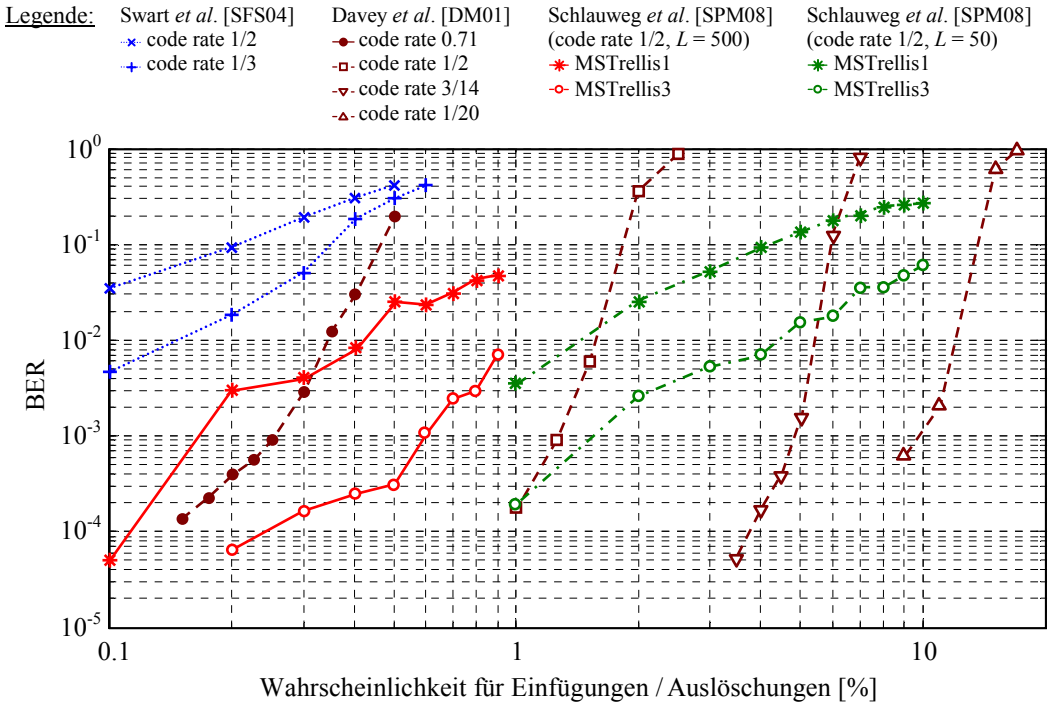


Abbildung 6.10 Vergleich der neu entwickelten Resynchronisation mit Verfahren anderer Autoren (Bitfehlerverhältnis aufgetragen über Wahrscheinlichkeit des Auftretens von Einfügungen /Auslöschungen), wobei $P_f = 0$

Die Kurven in Abbildung 6.10 zeigen, dass die Leistungsfähigkeit des von Swart *et al.* entwickelten Super-Trellis-Decoders durch die Modifikationen im Rahmen dieser Arbeit deutlich gesteigert werden konnte.

Verglichen mit dem Verfahren von Davey und MacKay erreicht die entwickelte Resynchronisation bei großen Blocklängen eine geringere Performance. Die Angaben in [DM01] beru-

hen auf Simulationen mit Blocklängen von mehreren tausend Bits. Laut Angaben der Autoren ist die Leistungsfähigkeit der IDS-Fehlerkorrektur bei kleinen Blocklängen eingeschränkt.

Um die gute Korrekturleistung der in dieser Arbeit entwickelten Resynchronisation bei kleinen Wasserzeichenlängen K bzw. Nachrichtenlängen L zu demonstrieren[77], wurden zusätzliche Simulationen durchgeführt. Dabei wurde die Länge der Sequenz auf $K = 32$ festgelegt (entsprechend dem in Abschnitt 5.3 entwickelten Blob-basierten Wasserzeichenverfahren mit 32 Daten-Blobs). Die so berechneten BER-Kurven zeigen für die verwendete Coderate $R_c = 1/2$ eine bessere Korrekturfähigkeit als das Verfahren von Davey und MacKay.

Im Rahmen dieser Arbeit wurden die Algorithmen zur Resynchronisation in der Programmierumgebung *MATLAB* nur für die Coderate $R_c = 1/2$ implementiert. Aufgabe zukünftiger Arbeiten könnte es sein, die entwickelte Resynchronisation auch für andere Coderaten einzusetzen.

6.4 Gesamtleistung des Blob-basierten Wasserzeichenverfahrens

Nachdem im vorherigen Abschnitt die Neuentwicklung einer Fehlerkorrektur für Einfügungen und Auslöschungen beschrieben wurde, findet in diesem Abschnitt die Leistungsanalyse des gesamten, um die Resynchronisation erweiterten Gray-Level-Blob-basierten Wasserzeichenverfahrens statt. Darüber hinaus geht dieser Abschnitt auch auf die Möglichkeit ein, beide neu entwickelten Wasserzeichenverfahren hintereinander für dasselbe Trägerbild anzuwenden und auf diese Weise zwei unabhängige Wasserzeichen in einem Bild einzubetten.

6.4.1 Robustheitsanalyse

Simulierte Robustheit der eingebetteten Wasserzeichendaten

Um die Robustheit des erweiterten Gray-Level-Blob-basierten Wasserzeichenverfahrens gegenüber Bildverarbeitungsoperationen zu ermitteln, wurden die gleichen Simulationen durchgeführt wie für das Verfahren ohne Resynchronisation (siehe Anhang G).

Je Testbild wurden $K + 1 = 33$ Gray-Level-Blobs mit den größten *LOG*-Amplitudenwerten ausgewählt. Nach einer Grundverstärkung G_{base} und einer Quantisierung der Blob-Amplituden (Einbetten der Wasserzeichenbits) wurden die durch Störungen hervorgerufenen Abweichungen der jeweiligen *LOG*-Amplitudenwerte berechnet. Dabei wurden während der Wasserzeichenextraktion eine Überlappung von $\tau = 0,05$ und damit die in Abschnitt 6.1.2 beschriebenen Kandidaten für Blob-Einfügungen und -Auslöschungen berücksichtigt.

[77] Bei der entwickelten Blob-basierten Wasserzeicheneinbettung handelt es sich um kleine Nachrichtenlängen.

Abbildung 6.11 zeigt einen Ausschnitt der Simulationsergebnisse aus Anhang H.2, vergleichbar mit den Bitfehlerverhältnissen der in Kapitel 5 entwickelten Blob-basierten Wasserzeicheneinbettung ohne Resynchronisation.

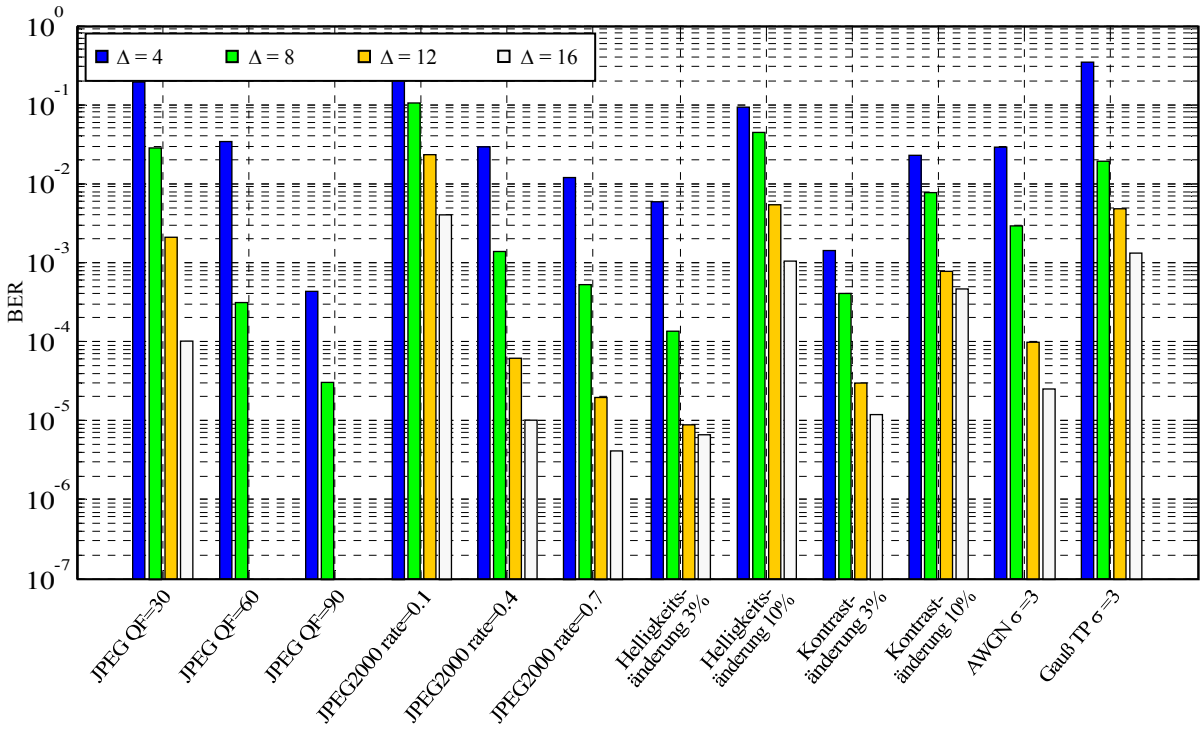


Abbildung 6.11 Robustheit des erweiterten Blob-basierten Wasserzeichenverfahrens gegenüber Kompression, Helligkeits- und Kontraständerung, Gaußschem Rauschen und Tiefpassfilterung bei unterschiedlichen Einbettungsstärken Δ (Parameter: $G_{base} = 10$, $\sigma_{\min} = 3$, $\sigma_{\max} = 5$).

Verglichen mit den Simulationsergebnissen des Wasserzeichenverfahrens ohne Resynchronisation (Abbildung 5.13, Seite 85) ist die Robustheit der eingebetteten Daten gegenüber Kompression, Helligkeits- und Kontraständerung oder Gaußschem Rauschen durch die in diesem Kapitel entwickelte Erweiterung nicht verbessert worden. Bezogen auf die Robustheit gegenüber geometrischen Veränderungen des Bildes wie Rotation, Translation oder Scherung ist jedoch eine deutliche Steigerung zu verzeichnen. Auch die Robustheit gegenüber Gaußscher Tiefpassfilterung und Bildschärfung (Unsharp-Masking) konnte durch die Erweiterung verbessert werden. Auf diese Weise ist es gelungen, eine gleichmäßige Robustheit der eingebetteten Daten gegenüber einer breiten Auswahl unterschiedlicher Störungen zu erreichen.

Vor allem durch die hinzugewonnene Robustheit gegenüber geometrischen Veränderungen des Trägerbildes, wie Rotation und Translation, wurde die Grundlage geschaffen, Wasserzeichendaten in Bildern einzubetten, die bspw. auch nach dem Ausdrucken/Einscannen des Bildes extrahiert werden können.

Trotz der aufgrund des Ausdruckens/Einscannens hervorgerufenen Kombination unterschiedlicher Störungen ist das in diesem Kapitel entwickelte Wasserzeichenverfahren in der Lage,

die beim Einbetten ausgewählten Gray-Level-Blobs während der Extraktion in den 52 Testbildern wieder zu finden. Die Störungen sind jedoch umfangreich (siehe Abbildung 5.1, S. 69) und stark abhängig von Mechanik und Optik der eingesetzten Drucker-/Scanner-Hardware.

Um das entwickelte Verfahren dahingehend zu optimieren, dass die eingebetteten Daten auch nach dem Ausdrucken/Einscannen des Trägerbildes korrekt ausgelesen werden können, sind zahlreiche Tests und Analysen mit handelsüblichen Druckern und Scannern notwendig. Diese weiterführenden Untersuchungen könnten Aufgaben für zukünftige Arbeiten sein.

Leistungsvergleich der entwickelten Verfahren

Die Leistungsfähigkeit der entwickelten Gray-Level-Blob-basierten Wasserzeicheneinbettung unterscheidet sich von der der adaptiven DWT-basierten Wasserzeicheneinbettung (Kapitel 3 und 4) in folgenden Punkten:

- der Aufwand zum Einbetten bzw. Extrahieren der Daten ist beim adaptiven DWT-basierten Verfahren geringer als beim Gray-Level-Blob-basierten Verfahren,
- die Menge der eingebetteten Daten (Kapazität) ist beim adaptiven DWT-basierten Verfahren größer,
- die durch das Gray-Level-Blob-basierte Verfahren eingebetteten Daten sind gegenüber einer breiteren Auswahl unterschiedlicher Störungen robuster.

Tabelle 6.2 stellt einen Vergleich beider Verfahren hinsichtlich der Robustheit der eingebetteten Daten mithilfe einer Bewertungsskala dar. Die Bewertungsskala ergibt sich nach folgendem Maßstab:

- 3 Haken (✓✓✓) entsprechen einem Bitfehlerverhältnis BER $\leq 10^{-4}$,
- 2 Haken (✓✓) entsprechen einem Bitfehlerverhältnis BER $\leq 10^{-2}$,
- 1 Haken (✓) entspricht einem Bitfehlerverhältnis BER $\leq 10^{-1}$,
- 1 Kreuz (✗) entspricht einem Bitfehlerverhältnis BER $> 10^{-1}$ (bzw. keiner Angabe).

Zusätzlich zu den beiden in dieser Arbeit entwickelten Wasserzeichenverfahren sind in Tabelle 6.2 drei weitere Verfahren aufgeführt. Es handelt sich dabei ebenfalls um Verfahren, mit deren Hilfe *Multi-Bit-Wasserzeichen*[78] robust gegenüber geometrischen Veränderungen des Trägerbildes eingebettet werden. Auch diese Verfahren benötigen zur Extraktion der Daten nicht das Originalbild.

[78] Wie bei den in dieser Arbeit entwickelten Verfahren, ist das korrekte Wasserzeichen auf Seiten der Extraktion nicht bekannt. *Zero-Bit-Wasserzeichenverfahren* hingegen setzen die Kenntnis des Wasserzeichens bei der Extraktion voraus.

Störung des Trägerbildes	adaptives, DWT-basiertes Verfahren	Gray-Level-Blob-basiertes Verfahren	log-polar/DFT-basiertes Verf. [FK05]	Harris-Laplace-basiertes Verf. [WW07]	Verf., basierend auf geom. Norm. [DBG+05]
JPEG-Kompression	✓✓✓	✓✓✓	✓✓	✓	✓✓✓
JPEG2000-Kompression	✓✓✓	✓✓✓	✗	✗	✗
AWGN	✓✓	✓✓	✗	✓	✓✓
Gauß-Tiefpass	✓✓	✓✓	✗	✓	✓✓✓
Helligkeitsänderung	✓✓	✓✓	✗	✗	✗
Kontraständerung	✓✓	✓✓	✗	✗	✗
Schärfung	✓✓	✓	✗	✓	✓
Spiegelung	✗	✓✓✓	✗	✗	✓✓✓
Skalierung	✓✓✓	✓✓	✓✓	✓	✓✓✓
Rotation	✗	✓✓✓	✓✓✓	✓	✓✓✓
Translation	✗	✓✓✓	✓✓✓	✓✓	✓✓✓
Scherung	✗	✓✓	✗	✓	✓✓✓
Cropping	✗	✓✓	✗	✓✓	✓✓

Tabelle 6.2 Bewertung der Robustheit der eingebetteten Daten der beiden in dieser Arbeit entwickelten Wasserzeichenverfahren im Vergleich mit Verfahren anderer Autoren

Die Angaben der Bitfehlerverhältnisse für die Leistungsfähigkeit der Verfahren der anderen Autoren sind den im Folgenden aufgeführten Veröffentlichungen entnommen.

- Das in [FK05] vorgestellte Wasserzeichenverfahren von Fung und Kunisa führt für das Trägerbild eine *Diskrete Fourier Transformation* (DFT) durch und bildet die DFT-Koeffizienten zum Einbetten der Daten in einer *log-polar*-Darstellung (LPM) ab. In dieser als LPM-DFT-Domain bezeichneten Darstellung können Wasserzeichendaten robust gegenüber RST-Störungen eingebettet werden. Das Verfahren zeigt zudem eine moderate Robustheit der eingebetteten Daten gegenüber einer JPEG-Kompression des Bildes.
- Das Wasserzeichenverfahren von Wang und Wu [WW07] basiert auf der Bestimmung bildinhaltsabhängiger Feature-Punkte mithilfe eines *Harris-Laplace-Detektors* und der Einbettung innerhalb kreisrunder, durch die Feature-Punkte positionierter, Bildregionen.
- Das Verfahren von Dong *et al*. [DBG+05] basiert auf einer geometrischen Normierung des Trägerbildes vor dem Einbetten der Wasserzeichendaten. Es zeichnet sich durch eine hohe Robustheit gegenüber einer breiten Auswahl unterschiedlicher Störungen aus.

Vergleichend ist festzustellen, dass das, im Rahmen dieser Arbeit entwickelte, Gray-Level-Blob-basierte Wasserzeichenverfahren eine ausgeglichene Robustheit gegenüber einer breiten Auswahl unterschiedlicher Störungen bietet.

6.4.2 Zusammenführung beider in dieser Arbeit entwickelter Verfahren

Die mit dem Gray-Level-Blob-basierten Wasserzeichenverfahren eingebetteten Daten sind robust gegenüber einer JPEG2000-Kompression des Trägerbildes. Aus diesem Grund kann das Gray-Level-Blob-basierte Verfahren, gefolgt vom JPEG2000-basierten Wasserzeichenverfahren, angewendet werden. Es können somit, wie bereits in [SPZM06] im Rahmen dieser Arbeit beschrieben, zwei voneinander unabhängige Wasserzeichen in einem Bild eingebettet werden (siehe Abbildung 6.12).

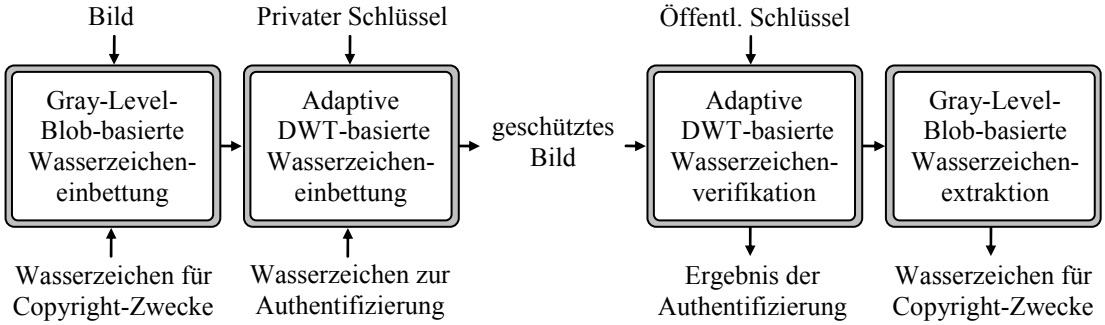


Abbildung 6.12 Sukzessive Anwendung zweier Wasserzeichenverfahren für dasselbe Trägerbild

Die beiden unabhängigen Wasserzeichen können für unterschiedliche Anwendungen eingesetzt werden.

- Mithilfe des JPEG2000-basierten Wasserzeichens (adaptive DWT-basierte Einbettung) kann die Echtheit des Trägerbildes überprüft werden (Authentifizierung).
- Das Gray-Level-Blob-basierte Wasserzeichenverfahren dient dazu, weitere Daten, bspw. für Copyright-Zwecke (siehe Abschnitt 2.1.2), einzubetten, die auch gegenüber geometrischen Veränderungen des Trägerbildes robust sind.

In einer Simulation wurden in die 52 Testbilder mithilfe des Gray-Level-Blob-basierten Wasserzeichenverfahrens zufällig generierte Wasserzeichen der Länge $K = 32$ Bit eingebettet[79]. Dieselben Bilder wurden anschließend jeweils mithilfe des JPEG2000-basierten Wasserzeichenverfahrens gegen Bild-Manipulationen geschützt (Einbettung eines zweiten Wasserzeichens der Länge $K = 1024$ Bit)[80]. Tabelle 6.3 zeigt das für das Gray-Level-Blob-basierte Wasserzeichenverfahren ermittelte Bitfehlerverhältnis. Es widerspiegelt den Einfluss der zweiten Wasserzeicheneinbettung durch das adaptive DWT-basierte Verfahren. Dieser Einfluss ist mit einer verlustbehafteten JPEG2000-Kompression der markierten Bilder vergleichbar.

[79] Die Einbettung des Gray-Level-Blob-basierten Wasserzeichens wurde für die Quantisierungsstärken $\Delta = 4$, $\Delta = 8$, $\Delta = 12$ und $\Delta = 16$ durchgeführt.

[80] Die Einbettung des JPEG2000-basierten Wasserzeichens wurde für die Quantisierungsstärken $\Delta_1 = 1$, $\Delta_1 = 2$, $\Delta_1 = 3$ und $\Delta_1 = 4$ durchgeführt, wobei $\Delta_2 = 3 \cdot \Delta_1$, $\tau = 1{,}5$ und $\alpha = 5$.

Blob-Grundverstärkung $G_{base} = 5$		Einbettungsstärke des adaptiven DWT-basierten Wasserzeichenverfahrens			
		$\Delta_1 = 1$ $\Delta_2 = 3$	$\Delta_1 = 2$ $\Delta_2 = 6$	$\Delta_1 = 3$ $\Delta_2 = 9$	$\Delta_1 = 4$ $\Delta_2 = 12$
Einbettungsstärke des Gray-Level-Blob-basierten Wasserzeichenverfahrens	$\Delta = 4$	0,0265	0,0781	0,1300	0,1628
	$\Delta = 8$	0,0043	0,0153	0,0423	0,0957
	$\Delta = 12$	0,0012	0,0036	0,0168	0,0333
	$\Delta = 16$	0,0002	0,0009	0,0051	0,0178

Blob-Grundverstärkung $G_{base} = 10$		Einbettungsstärke des adaptiven DWT-basierten Wasserzeichenverfahrens			
		$\Delta_1 = 1$ $\Delta_2 = 3$	$\Delta_1 = 2$ $\Delta_2 = 6$	$\Delta_1 = 3$ $\Delta_2 = 9$	$\Delta_1 = 4$ $\Delta_2 = 12$
Einbettungsstärke des Gray-Level-Blob-basierten Wasserzeichenverfahrens	$\Delta = 4$	0,0404	0,0818	0,1422	0,1568
	$\Delta = 8$	0,0039	0,0129	0,0405	0,0770
	$\Delta = 12$	0,0007	0,0032	0,0108	0,0294
	$\Delta = 16$	0,0002	0,0005	0,0026	0,0082

Tabelle 6.3 Simuliertes Bitfehlerverhältnis für die Wasserzeichenextraktion durch das Gray-Level-Blob-basierte Verfahren nach der Einbettung des adaptiven DWT-basierten Wasserzeichens im selben Trägerbild, wobei die Grundverstärkung der Blobs $G_{base} = 5$ (links) bzw. $G_{base} = 10$ (rechts) gewählt wurde

6.5 Fazit

In den vorhergehenden Abschnitten dieses Kapitels wurde das Gray-Level-Blob-basierte Wasserzeichenverfahren um eine Resynchronisation zur Korrektur von Einfügungen und Auslöschungen erweitert. Dazu wurden die Algorithmen eines Viterbi-Decoders verändert, mit dem das während der Einbettung codierte Wasserzeichen bei der Extraktion decodiert wird.

Es wurde gezeigt, dass Einfügungen und Auslöschungen von durch den Feature-Detektor ausgewählten Blobs mithilfe der entwickelten Resynchronisation erkannt und korrigiert werden können. Auf diese Weise konnte die Leistungsfähigkeit des vorgestellten Gray-Level-Blob-basierten Wasserzeichenverfahrens vor allem hinsichtlich geometrischer Veränderungen des Trägerbildes erheblich gesteigert werden.

Weiterhin wurde die sukzessive Anwendung der beiden in dieser Arbeit entwickelten Wasserzeichenverfahren für ein und dasselbe Trägerbild untersucht. Es wurde gezeigt, dass die Einbettung des JPEG2000-basierten Wasserzeichens ein zuvor im selben Bild eingebettetes Gray-Level-Blob-basiertes Wasserzeichen nur geringfügig stört.

Kapitel 7

Zusammenfassung und Ausblick

7.1 Zusammenfassung

Das Ziel dieser Dissertation bestand in der Entwicklung eines neuen Wasserzeichensystems zur effizienten und manipulationssicheren Überprüfung der Echtheit von Bildern.

Um Bilder bereits direkt während der Aufnahme in der Kamera durch ein Wasserzeichen vor unerlaubten bildinhaltlichen Veränderungen zu schützen, wurde ein Wasserzeichensystem entwickelt, das in den Prozess einer JPEG2000-Bildkompression integriert ist.

Es wurde erkannt, dass durch die quantisierten Koeffizienten der Diskreten Wavelet-Transformation (DWT) eines Bildes eine Digitale Signatur zur Überprüfung der Echtheit (Authentifizierung) gebildet werden kann. Anhand umfangreicher Simulationen wurden diejenigen DWT-Koeffizienten bestimmt, die den Bildinhalt manipulationssicher und gleichfalls robust gegenüber erlaubten Bildnachbearbeitungsoperationen (semi-fragil) repräsentieren.

Das entwickelte System nutzt eine, an die Eigenschaften der JPEG2000-Bildkompression angepasste, Quantisierung mit Totzone, um die gebildete Digitale Signatur robust und nichtwahrnehmbar im Bild als Wasserzeichen einzubetten. Auf diese Weise kann sie auch bei einer erlaubten Nachbearbeitung oder Konvertierung des Bildes in ein anderes Kompressionsformat nicht verloren gehen.

Durch den Einsatz einer Fehlerkorrektur, die nicht nur die eingebetteten Wasserzeichendaten, sondern auch die Generierung der vom Bildinhalt abhängenden Digitalen Signatur umfasst, konnte die Robustheit gegenüber erlaubten Bildoperationen deutlich erhöht werden. Es wurde gezeigt, dass die Manipulationssicherheit des Systems dadurch nicht eingeschränkt wird.

Zur Erweiterung der Robustheit gegenüber erlaubten Bildoperationen wurde für die neue semi-fragile JPEG2000-Bildauthentifizierung eine Helligkeits- und Kontrastnormierung entwickelt. Darüber hinaus wurde die Wasserzeichengenerierung und -einbettung, in Analogie zur Bildunterteilung des JPEG2000-Kompressionsstandards, an die Größe des Bildes angepasst. Damit kann die Echtheit eines Bildes unabhängig von der Bildgröße verifiziert werden.

Um die Leistung weiterhin zu steigern, wurde eine Adaption der zur Wasserzeichengenerierung und -einbettung benutzten Quantisierungsschrittweite an die lokalen Texturmerkmale des Bildes entwickelt.

Diese Adaption muss bei Einbettung und Extraktion des Wasserzeichens identisch sein. Um Abweichungen zu korrigieren, wurde die Adaption auf Seiten der Extraktion durch eine Wichtung mit Informationen der texturbasierten Bildsegmentierung erweitert. Eine Vorverzerrung der Texturmerkmale kann somit zu Gunsten der Nicht-Wahrnehmbarkeit der durch die Wasserzeicheneinbettung verursachten Trägerbildverzerrungen entfallen.

Die Leistungsfähigkeit der neu entworfenen JPEG2000-basierten Bildauthentifizierung wurde ausführlich analysiert und mit der Leistungsfähigkeit ähnlicher Verfahren anderer Autoren verglichen. Dabei hat sich herausgestellt, dass das neue Authentifizierungssystem den anderen Verfahren deutlich überlegen ist.

Aufgrund der gewonnenen Erkenntnisse wurde in dieser Arbeit ein zweites Wasserzeichenverfahren entwickelt, durch welches, im Gegensatz zu dem JPEG2000-basierten Verfahren, Daten auch robust gegenüber geometrischen Veränderungen des Trägerbildes eingebettet werden können. Dieses auf der Veränderung so genannter Gray-Level-Blobs im Gaußschen Skalenraum basierende Wasserzeichenverfahren wurde dahingehen konzipiert, die eingebetteten Daten auch nach einer Rotation, Verschiebung, Scherung oder dem Entfernen von Spalten oder Zeilen des Trägerbildes extrahieren zu können.

Mithilfe von Simulationen wurden die Einflüsse einer Vielzahl von Bildverarbeitungsoperationen auf die Amplitudenwerte der Gray-Level-Blobs unterschiedlicher Bilder analysiert. Es wurde gezeigt, dass Wasserzeichendaten effizient und nicht-wahrnehmbar durch die Quantisierung dieser Amplitudenwerte eingebettet werden können.

Im Laufe der Untersuchungen wurde erkannt, dass sich die Positionen der zur Einbettung ausgewählten Gray-Level-Blobs infolge von Störungen geringfügig ändern können. Es kann dadurch zu Einfügungen und Auslöschungen von Gray-Level-Blobs während der Wasserzeichenextraktion und somit zur Desynchronisation innerhalb der eingebetteten Daten kommen.

Zur Resynchronisation der auftretenden Einfügungen und Auslöschungen wurde eine Modifikation des zur Decodierung von Faltungscodes eingesetzten Viterbi-Decoders durchgeführt. Es wurde gezeigt, dass durch den modifizierten Viterbi-Decoder Einfügungen und Auslöschungen von Gray-Level-Blobs erkannt und korrigiert werden können. Auf diese Weise wurde die Robustheit des entwickelten Wasserzeichenverfahrens deutlich gesteigert.

Auch die Leistungsfähigkeit des neuen Gray-Level-Blob-basierten Wasserzeichenverfahrens wurde ausführlich analysiert und mit der Leistungsfähigkeit ähnlicher Verfahren anderer Au-

toren verglichen. Hervorzuheben ist vor allem die ausgeglichene Robustheit der eingebetteten Daten gegenüber einer breiten Auswahl unterschiedlicher Störungen. Zu diesen Störungen zählen auch geometrische Veränderungen des Trägerbildes.

7.2 Ausblick auf zukünftige Arbeiten

Das im Rahmen dieser Dissertation entwickelte JPEG2000-basierte Wasserzeichenverfahren zur Überprüfung der Echtheit von Bildern wurde im Zuge von Projektarbeiten zu Demonstrationszwecken auch für die Plattform Pocket PC (Betriebssystem: Windows) implementiert. Nachdem die Leistungsfähigkeit des vorgestellten Systems in dieser Arbeit ausführlich analysiert wurde, wäre in zukünftigen Arbeiten eine Integration in ein echtes Kameramodell wünschenswert. Auf diese Weise könnte der Prozess der Wasserzeichengenerierung und -einbettung in eine abgeschlossene, sichere Umgebung versetzt werden. Dadurch könnte ein insgesamt fälschungssicheres Authentifizierungssystem für Bilder erzeugt werden.

Mithilfe des Gray-Level-Blob-basierten Wasserzeichenverfahrens wurde die Grundlage geschaffen, um Daten sehr robust gegenüber einer breiten Auswahl unterschiedlicher Störungen in Bilder einzubetten. Zu diesen Störungen zählen auch geometrische Veränderungen des Trägerbildes. Das entwicklete Verfahren kann in einer zukünftigen Entwicklungsstufe ebenfalls dazu eingesetzt werden, um die Echtheit von Bildern zu überprüfen. Eine Digitale Signatur könnte dazu, wie bereits in [SM09] beschrieben, anhand der Gray-Level-Blob-Amplitudenwerte gebildet werden. Diese erwiesen sich in Untersuchungen als sehr robust gegenüber zahlreichen erlaubten Bildoperationen. Wird der Bildhalt unerlaubt verändert, zeigt sich die Auswahl der Gray-Level-Blobs (gewollt) als deutlich fragil. Exakte Untersuchungen diesbezüglich waren nicht mehr Gegenstand dieser Dissertation und können Thema zukünftiger Arbeiten sein.

Auch die im Rahmen dieser Arbeit entwickelten Algorithmen zur Resynchronisation können in zukünftigen Arbeiten erweitert werden. Sie wurden zu Testzwecken in der Programmierumgebung *MATLAB* für die Coderate $R_c = 1/2$ implementiert, um die Funktionsfähigkeit unter Beweis zu stellen. Ein Einsatz für andere Coderaten wäre ohne weit reichende Änderungen möglich.

Literatur

[BB04] Barni, M. and Bartolini, F., Watermarking Systems Engineering - Enabling Digital Assets Security and Other Applications, Marcel Dekker Inc., New York, USA, 2004, ISBN: 0-8247-4806-9.

[BBCP98] Barni. M., Bartolini, F., Cappellini, V., and Piva, A., A DCT-Domain System for Robust Image Watermarking, Signal Processing, 66 (3), May 1998, pp. 357-372.

[BCM00] Bas, P., Chassery, J.-M., and Macq, B., Robust Watermarking Based on the Warping of Pre-defined Triangular Patterns, In Proc. of SPIE, 3971, Jan. 2000, San Jose, USA, pp. 99-109.

[BCM02] Bas, P., Chassery, J.-M., and Macq, B., Geometrically Invariant Watermarking Using Feature Points, IEEE Transactions on Image Processing, 11 (9), Sept. 2002, pp. 1014-1028.

[BGML96] Bender, W., Gruhl, D., Morimoto, N., and Lu, A., Techniques for Data Hiding, IBM Systems Journal, 35 (3&4), 1996, pp. 313-336.

[Bos98] Bossert, M., Kanalcodierung, 2. Auflage, Verlag B.G. Teubner, Stuttgrat, Germany, 1998, ISBN: 3-519-16143-5.

[Buc03] Buchmann, J., Einführung in die Kryptographie, Springer Verlag, Berlin, Germany, 2003, ISBN: 3-540-40508-9.

[CKLS97] Cox, I. J., Kilian, J., Leighton, T., and Shamoon, T., Secure Spread Spectrum Watermarking for Multimedia, IEEE Transactions on Image Processing, 6 (12), 1997, pp. 1673-1687.

[CM02] Cox, I. J. and Miller, M. L., Electronic Watermarking: The First 50 Years, EURASIP Journal on Applied Signal Processing, 2002 (2), 2002, pp. 126-132.

[CMB+08] Cox, I. J., Miller, M. L., Bloom, J. A., Fridrich, J., and Kalker, T., Digital Watermarking and Steganography, 2. Edition, Morgan Kaufmann Publishers, Burlington, USA, 2008, ISBN: 978-0-12-372585-1.

[CS08] Coumou, D. J. and Sharma, G., Insertion, Deletion Codes with Feature-Based Embedding: A New Paradigm for Watermark Synchronization with Applications to Speech Watermarking, In IEEE Transactions on Information Forensics and Security, 3 (2), 2008, pp. 153-165.

[CW01] Chen, B. and Wornell, G.W., Quantization index modulation: A Class of Provably Good Methods for Digital Watermarking and Information Embedding, IEEE Transactions on Information Theory, 47, May 2001, pp. 1423-1443.

[DBG+05] Dong, P., Brankov, J. G., Galatsanos, N. P., Yang, Y., and Davoine, F., Digital Watermarking Robust to Geometric Distortions, IEEE Transactions on Image Processing, 14 (12), 2005, pp. 2140-2150.

[DFS00] Dittmann, J., Fiebig, T., and Steinmetz, R., A New Approach for Transformation Invariant Image and Video Watermarking in the Spatial Domain: SSP - Self-Spanning Patterns, In Proc. of SPIE, 3971, Jan. 2000, San Jose, USA, pp. 176-185.

[DM01] Davey, M. C. and MacKay, D. J. C., Reliable Communication over Channels with Insertions, Deletions and Substitutions, IEEE Transactions on Information Theory, 47 (2), 2001, pp. 687-698.

[EG01] Eggers, J. J. and Girod, B., Blind Watermarking Applied to Image Authentication, In Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing, May 2001, Salt Lake City, USA, pp. 1977-1980.

[EG02] Eggers, J. J. and Girod, B., Informed Watermarking, Kluwer Academic Publishers, Norwell, USA, 2002, ISBN: 1-4020-7071-3.

[ESG00] Eggers, J. J., Su, J., and Girod, B., A Blind Watermarking Scheme Based on Structured Codebooks, In Proc. of IEE Seminar on Secure Images and Image Authentication, 4, London, UK, April 2000, pp. 1-6.

[ESC+04] Ekici, Ö., Sankur, B., Coşkum B., Nazi U., and Akcay, M., Comparative evaluation of semifragile watermarking algorithms, Journal of Electronic Imaging, 13, Jan. 2004, pp. 209-216.

[Far06] Farid, H., Exposing Digital Forgeries in Scientific Images, In Proc. of ACM Multimedia and Security Workshop, Sept. 2006, Geneva, Switzerland, pp. 29-36.

[FK05] Fung, W.W.L. and Kunisa. A., Rotation, Scaling, and Translation-Invariant Multi-Bit Watermarking Based on LOG-Polar Mapping and Discrete Fourier Transform, In Proc. of IEEE International Conference on Multimedia and EXPO, July 2005, pp. 1-4.

[FKK06] Fei, C., Kundur, D., and Kwong, R. H., Analysis and Design of Secure Watermark-Based Authentication Systems, IEEE Transactions on Information Forensics and Security, 1 (1), March 2006, pp. 43-55.

[For66] Forney, G.D., Concatenated Codes, The MIT Press, Cambrigde, USA, 1966.

[Fri93] Friedman, G. L., The Trustworthy Digital Camera: Restoring Credibility to the Photographic Image, IEEE Transactions con Consumer Electronics, 39 (4), Sept. 1993, pp. 905-910.

[Fri99] Fridrich, A Hybrid Watermark for Tamper Detection in Digital Images, In Proc. of International Symposium on Signal Processing and its Applications, Aug. 1999 Brisbane, Australia, pp. 301-304.

[HJJH01] Huang, D., Jiufen, L., Jiwu, H., and Hongmei, L., A DWT-Based Image Watermarking Algorithm, In Proc. of IEEE International Conference on Multimedia and Expo, Aug. 2001, Tokyo, Japan, pp. 313-316.

[HK99] Hartung, F. and Kutter, M., Multimedia Watermarking Techniques, In Proceedings of the IEEE, 87 (7), July 1999, pp. 1079-1107.

[ISO00] ISO/IEC 15444-1 JTC1/SC29/WG1/N1890R, (ISO/IEC FDIS15444-1), JPEG2000 - Part I Final Draft International Standard, Sept. 2000.

[ISO06] ISO/IEC 15444-8 JTC1/SC29/WG1/N3853, (ISO/IEC FDIS15444-8), JPSEC - Part 8 Final Draft International Standard, Feb. 2006.

[JF05] Johnson, M. K., Farid, H., Exposing Digital Forgeries by Detecting Inconsistencies in Lightning, In Proc. of 7th ACM Multimedia and Security Workshop, Aug. 2005, New York, USA, pp. 1-10.

[JF06] Johnson, M. K., Farid, H., Exposing Digital Forgeries through Chromatic Aberration, In Proc. of ACM Multimedia and Security Workshop, Sept. 2006, Geneva, Switzerland, pp. 48-55.

[KBE99] Kutter, M., Bhattacharjee, S. K., and Ebrahimi, T., Towards Second Generation Watermarking Schemes, In Proc. of IEEE International Conference on Image Processing, 1, Oct. 1999, Kobe, Japan, pp. 320-323.

[KH99] Kundur, D. and Hatzinakos, D., Digital Watemarking for Telltale Tamper-Proofing and Authentication, In Proceedings of the IEEE - Special Issue on Identification of Multimedia Information, 87 (7), July 1999, pp. 1167-1180.

[KM01] Kirovski, D. and Malvar, H., Robust Covert Communications over a Public Audio Channel using Spread Spectrum, In Proc. of 4th Information Hiding Workshop, Pittsburgh, USA, April 2001, pp. 354-368.

[KXYM07] Kim, H.-J., Xiang, S., Yeo, I. K., and Maitra, S., Robustness Analysis of Patchwork Watermarking Schemes, Chapter VIII in Digital Audio Watermarking Techniques and Technologies: Applications and Benchmarks, (ed.) Cvejik, N. and Seppanen, T., Idea Group Publishing, Hershey, USA, 2007, ISBN: 978-1-59904-513-9.

[LC98] Lin, C. Y. and Chang, S.-F., A Robust Image Authentication Method Surviving JPEG Lossy Compression, In Proc. of SPIE, 3312, Jan. 1998, San Jose, USA, pp. 296-307.

[LC00] Lin, C. Y. and Chang, S.-F., Semi-Fragile Watermarking for Authenticating JPEG Visual Content, In Proc. of SPIE, 3971, Jan. 2000, San Jose, USA, pp. 140-151.

[LFG06] Lukáš, J., Fridrich, J., and Goljan, M., Digital Camera Identification from Sensor Pattern Noise, IEEE Transactions on Information Security and Forensics, 1(2), June 2006, pp. 205-214.

[LHX01] Liu, H.-M., Huang, J.-W., and Xiao, Z.-M., An Adaptive Video Watermarking Algorithm, In Proc. of IEEE International Conference on Multimedia and Expo, Aug. 2001, Tokyo, Japan, pp. 257-260.

[Lin94] Lindeberg, T., Scale-Space Theory in Computer Vision, Kluver Academic Publishers, Dortrecht, Netherlands, 1994, ISBN: 0-7923-9418-6.

[Lin98] Lindeberg, T., Feature Detection with Automatic Scale Selection, International Journal of Computer Vision, 30 (2), November 1998, pp. 77-116.

[LKL06] Lee, H.-Y., Kim, H., and Lee, H.-K., Robust Image Watermarking Using Local Invariant Features, In Proc. of SPIE, 45 (3), March 2006, pp.037002:1-11.

[LLL07] Lee, H.-Y., Lee, C.-H., and Lee, H.-K., Geometrically Invariant Watermarking: Synchronization Trough Circular Hough Transform, In Multimedia Tools Applications, 34 (3), Sept. 2007, pp. 337-353.

[LPD00] Lin, E. T., Podilchuk, C. I., and Delp, E. J., Detection of Image Alterations using Semi-Fragile Watermarks, In Proc. of SPIE, 3971, Jan. 2000, San Jose, USA, pp. 152-163.

[LSC05] Lu, C.-S., Sun, S.-W., and Chang, P.-C., Robust Hash-Based Image Watermarking with Resistance to Geometric Distortions and Watermark-Estimation Attack, In Proc. of SPIE, 5681, Jan. 2005, San Jose, USA, pp. 147-163.

[MO98] Maes, M. J. J. J. and van Overveld, C. W. A. M., Digital Watermarking by Geometric Warping, In Proc. of IEEE International Conference on Image Processing, 2, Oct. 1998, Chicago, USA, pp. 424-426.

[MSCS06] Maeno, K., Sun, Q., Chang, S.-F., and Suto, M., New Semi-Fragile Image Authentication Watermarking Technique Using Random Bias and Nonuniform Quantization, IEEE Transactions on Multimedia, 8 (1), Feb. 2006, pp. 32-45.

[MT01] Mansour, M. F. and Tewfik, A. H., Efficient Decoding of Watermarking Schemes in the Presence of False Alarms, In Proc. of IEEE Workshop on Multimedia Signal Processing, Cannes, France, October 2001, pp. 523-528.

[NSA+04] Ni, Z., Shi, Y. Q., Ansari, N., Su, W., Sun, Q., and Lin, X., Robust Lossless Image Data Hiding, In Proc. of IEEE International Conference on Multimedia and Expo, 3, June 2004, Taipei, Taiwan, pp. 2199-2202.

[PMBA04] Pérez-González, F., Mosquera, C., Barni, M., and Abrardo, A., Rational Dither Modulation: A Novel Data-Hiding Method Robust to Value-Metric Scaling Attacks, In Proc. of 6th IEEE Workshop on Multimedia Signal Processing, Sept. 2004, Siena, Italy, pp. 139-142.

[PS98] Papadopoulos, H. C. and Sundberg, C.-E. W., Simultaneous Broadcasting of Analog FM and Digital Audio Signals by means of Precanceling Techniques, IEEE Transactions on Communications, 46 (9), September 1998, pp. 1233-1242.

[PSM06] Pröfrock, D., Schlauweg, M., and Müller, E., Content-Based Watermarking by Geometric Warping and Feature-Based Image Segmentation, In Proc. of ACM/ IEEE International Conference on Signal-Image Technology & Internet-Based Systems, Hammamet, Tunisia, Dec. 2006, pp. 572-581.

[PZ98] Podilchuk, C. I. and Zeng, W., Image-Adaptive Watermarking Using Visual Models, IEEE Journal on Special Areas in Communications, 16 (4), May 1998, pp. 525-539.

[Que01] Queluz, M. P., Authentication of Digital Images and Video: Generic Models and a New Contribution, Signal Processing - Image Communication, 216 (5), Jan. 2001, pp. 461-475.

[RD02] Rey, C. and Dugelay, J.-L., A Survey of Watermarking Algorithms for Image Authentication, EURASIP Journal of Applied Signal Processing, 6, March 2002, pp. 613-621.

[SAL+89] Schreiber, W. F., Adelson, E. H., Lippman, A. B., Rongshu, G., Monta, P., Popat, A., Sallic, H., Shen, P., Tom, A., Zangi, K., and Netravali, A. N., A Compatible High-Definition Television System using the Noise-Margin Method of Hiding Enhanced Information, SMPTE Journal, 98 (12), December 1989, pp. 873-879.

[SC05] Sun, Q. and Chang, S. F., A Secure and Robust Digital Signature Scheme for JPEG2000 Image Authentication, IEEE Transactions on Multimedia, 7 (3), June 2005, pp. 480-494.

[SCE01] Skodras, A., Christopoulis, C., and Ebrahimi, T., The JPEG 2000 Still Image Compression Standard, IEEE Signal Processing Magazine, 18 (5), Sept. 2001, pp. 36-58.

[SFS04] Swart, T. G., Ferreira, H. C., and dos Santos M.P.F., Using Parallel-Interconnected Viterbi-Decoders to Correct Insertion/Deletion Errors, In Proc. of IEEE AFRICON Conference in Africa, Garborone, Botswana, 2004, pp. 341-344.

[SJM+04] Solanki, K., Jacobsen, N., Madhow, U., Manjunath, B. S., and Chandrasekaran, S., Robust Image-Adaptive Data Hiding Using Erasures and Error Correction, IEEE Transactions on Image Processing, 13 (12), December 2004, pp. 1627-1639.

[SM09] Schlauweg, M. and Müller, E., Gaussian Scale-Space Features for Semi-Fragile Image Authentication, In Proc. of 27th Picture Coding Symposium, May 2009.

[SNP00] Solachidis, V., Nikolaidis, N., and Pitas, I., Watermarking Polygonal Lines Using Fourier Descriptors, In Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing, 6, June 2000, Istanbul, Turkey, pp. 1955-1958.

[SPM06] Schlauweg, M., Pröfrock, D., and Müller, E., JPEG2000-Based Secure Image Authentication. In Proc. of ACM Multimedia and Security Workshop, Sept. 2006, Geneva, Switzerland, pp. 62-67.

[SPM07a] Schlauweg, M., Pröfrock, D., and Müller, E., Avoiding Hard Decisions in Adaptive Watermarking, In Proc. of 14th IEEE International Conference on Image Processing, Sept. 2007, San Antonio, USA, pp. 453-456.

[SPM07b] Schlauweg, M., Pröfrock, D., and Müller, E., Soft Feature-Based Watermark Decoding with Insertion/Deletion Correction, In Proc. of Information Hiding Workshop, June 2007, Saint Malo, France, pp. 236-250.

[SPM08] Schlauweg, M., Pröfrock, D., and Müller, E., Correction of Insertions and Deletions in Selective Watermarking, In Proc. of 4th IEEE International Conference on Signal-Image Technology & Internet-Based Systems, Nov. 2008, Bali, Indonesia, pp. 277-284.

[SPPM05a] Schlauweg, M., Pröfrock, D., Palfner, T., and Müller, E., Quantization-Based Semi-fragile Public-key Watermarking for Secure Image Authentication. In Proc. of SPIE, 5915, July 2005, pp. 41-51.

[SPPM05b] Schlauweg, M., Palfner, T., Pröfrock, D., and Müller, E., The Achilles' Heel of JPEG-Based Image Authentication, in Proc. of IASTED International Conference on Communication, Network and Information Security, Nov. 2005, Phoenix, USA, pp. 1-6.

[SPZM06] Schlauweg, M., Pröfrock, D., Zeibich, B., and Müller, E., Dual Watermarking for Protection of Rightful Ownership and Secure Image Authentication, In Proc. of

14th ACM Multimedia - Workshop on Content Protection and Security, Oct. 2006, Santa Barbara, USA, pp. 56-63.

[SPZM08] Schlauweg, M., Pröfrock, D., Zeibich, B., and Müller, E., Self-Synchronizing Robust Texel Watermarking in Gaussian Scale-Space, In Proc. of 10th ACM Workshop on Multimedia and Security, Sept. 2008, Oxford, UK, pp. 53-61.

[Str05] Strutz, T., Bilddatenkompression. Grundlagen, Codierung, Wavelets, JPEG, MPEG, H.264, 3. Auflage, Verlag Vieweg+Teubner, Wiesbaden, Germany, 2005, ISBN: 3-528-23922-0.

[SY04] Seo, J. and Yoo, C. D., Image Watermarking Based on Scale-Space Representation, In Proc. of SPIE, 5306, Jan. 2004, San Jose, USA, pp. 560-570.

[SZTB98] Swanson, M. D., Zhu, B., Tewfik, A. H., and Boney, L., Robust Audio Watermarking Using Perceptual Masking, Signal Processing, 66 (3), May 1998, pp. 337-355.

[TD97] Tao, B. and Dickinson, B., Adaptive Watermarking in the DCT Domain, In Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing, 4, Apr. 1997, Munich, Germany, pp. 2985-2988.

[TH03] Tang, C.-W. and Hang, M.-H., A Feature-Based Robust Digital Image Watermarking Scheme, IEEE Transactions on Signal Processing, 51 (4), Apr. 2003, pp. 950-959.

[TM02] Taubman, D. S. and Marcellin, M. W., JPEG2000: Image Compression Fundamentals, Standards and Practice, Kluver Academic Publishers, 2002.

[VBH+01] Van der Veen, M, Bruekers, F., Haitsma, J., Kalker, T., Lemma, A. N., and Oomen, W., Robust, Multi-Functional and High-Quality Audio Watermarking Technology, In Proc. of Audio Engineering Society Convention, 110 (5345), May 2001, pp. 1-9.

[Vit67] Viterbi, A. J., Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm, IEEE Transactions on Information Theory, 13 (2), April 1967, pp. 260-269.

[VS06] Vielhauer, C. and Schott, M., Image Annotation Watermarking: Nested Object Embedding using Hypergraph Model, In Proc. of 8th ACM Multimedia and Security Workshop, Geneva, Switzerland, September 2006, pp. 182-189.

[VVK+05] Vila-Forcén, J. E., Voloshynovskiy, S., Koval, O., Pun, T., and Pérez-González, F., Worst Case Additive Attack against Quantization-Based Data-Hiding Methods, In Proc. of SPIE, 5681, January 2005, San Jose, USA, pp. 136-146.

[WPD99] Wolfgang, R. B., Podilchuk, C. I., and Delp, E. J., Perceptual Watermarks for Digital Images and Video, In Proc. of IEEE - Special Issue on Identification and Protection of Multimedia, 87 (7), July 1999, pp. 1108-1126.

[WW07] Wang, X.-Y. and Wu, J., A Feature-Based Robust Digital Image Watermarking Against Desynchronization Attacks, IEEE International Journal of Automation and Computing, 4 (4), Oct. 2007, pp. 428-432.

[WXQ06] Weinheimer, J., Xi, Q., and Qi, J., Towards a Robust Feature-Based Watermarking Scheme, In Proc. of IEEE International Conference on Image Processing, Oct. 2006, Atlanta, USA, pp. 1401-1404.

[XZHL06] Xiong, S., Zhou, J., He, K., and Lang, F., A Multipurpose Image Watermarking Method Based on Adaptive Quantization of Wavelet Coefficients, In Proc. of IEEE International Multi-Symposiums on Computer and Computational Science, June 2006, Hangzhou, China, pp. 294-297.

[YLL01] Yu, G.-W., Lu, C.-S., and Liao, H.-Y. M., Mean Quantization-Based Fragile Watermarking for Image Authentication, Optical Engineering, 40 (7), July 2001, pp. 1396-1408.

[ZLZS07] Zheng, D., Liu, Y., Zhao, J., and Saddik, E., A Survey of RST Invariant Image Watermarking Algorithms, In ACM Computing Surveys, 39 (2-5), June 2007.

[ZQS+04] Zhang, Z., Qui, G., Sun, Q., Lin, X., Ni, Z., and Shi, Y. Q., A Unified Authentication Framework for JPEG2000, In Proc. of IEEE International Conference on Multimedia and Expo, 2, June 2004, Taipei, Taiwan, pp. 915-918.

[ZST04] Zhu, B. B., Swanson, M. D., and Tewfik, A. H., When seeing isn't believing, IEEE Transactions on Signal Processing, 21, March 2004, pp. 40-49.

Anhang A

Verwendete Testbilder

In dieser Arbeit wurden die im Folgenden aufgeführten 52 Testbilder verwendet.

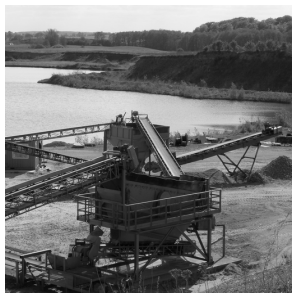
Allgemeine Daten

- Bildgröße: 512 × 512 Pixel
- Farbtiefe: 8 Bit (Graustufen)

Name: tb01

Name: tb02

Name: tb03



Name: tb04

Name: tb05

Name: tb06

Name: tb07

Name: tb08

Name: tb09

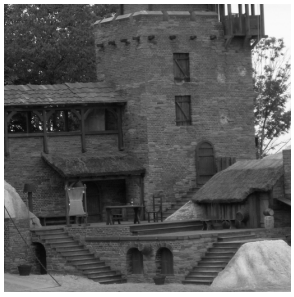
Name: tb10

Name: tb11

Name: tb12

Name: tb13

Name: tb14



Name: tb15

Name: tb16

Name: tb17

Name: tb18

Name: tb19

Name: tb20

Name: tb21

Name: tb22

Name: tb23

Name: tb24

Name: tb25

Name: tb26

Name: tb27

Name: tb28

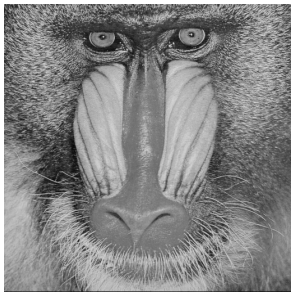
Name: tb29

Name: tb30

Name: tb31

Name: tb32

Name: tb33



Name: tb34

Name: tb35

Name: tb36

Name: tb37

Name: tb38

Name: tb39

Name: tb40

Name: tb41

Name: tb42

Name: tb43

Name: tb44

Name: tb45

Name: tb46

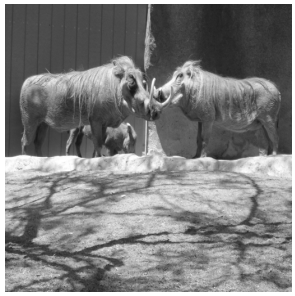
Name: tb47

Name: tb48

Name: tb49

Name: tb50

Name: tb51



Name: tb52

Anhang B

Analyse der Diskreten Wavelet-Transformationskoeffizienten eines Bildes

Sollen die Koeffizienten der Diskreten Wavelet-Transformation (DWT) eines Bildes zum Einbetten von Wasserzeichendaten verwendet werden, ist es von Interesse, wie sich die Koeffizienten als Träger der eingebetteten Daten infolge von Störungen[81] verändern. Darüber hinaus ist es wichtig zu wissen, welche DWT-Koeffizienten (Subbänder) sich aufgrund der Störungen am wenigsten ändern und damit zum robusten Einbetten geeignet sind. In diesem Kapitel des Anhangs wird die im Kompressionsstandard JPEG2000 zum Einsatz kommende Transformationsdomain DWT hinsichtlich dieser Fragen analysiert.

B.1 Veränderung der DWT-Koeffizienten infolge von Störungen

In einer Simulation wurden für die 52 verwendeten Testbilder jeweils die DWT-Koeffizienten für die Zerlegungsstufen $Z = 1$ bis $Z = 4$ berechnet (vgl. Abschnitt 2.2.2). Nach der Anwendung von Störungen mit unterschiedlichen Parametern wurden die DWT-Zerlegungen ein zweites Mal durchgeführt. Die Veränderungen der Koeffizienten (ohne Helligkeits-/Kontrastnormierung) aufgrund der Störungen sind im Abschnitt B.1.1 für alle Bilder zusammengefasst dargestellt. Dabei wird zwischen den LL-Band- und den HL-, LH-, HH-Band-Koeffizienten unterschieden. Letztere sind gemeinsam abgebildet, da die Auswirkungen der Störungen für das HL-, LH-, HH-Band sehr ähnlich sind. Die Veränderungen der Koeffizienten sind in Form von Wahrscheinlichkeitsdichteverteilungen[82] $pdf\left(x-\hat{x}\right)$ dargestellt, wobei x einen DWT-Koeffizienten vor und $\hat{x}$ denselben Koeffizienten nach der Störung bezeichnet.

In den Abbildungen des Abschnittes B.1.2 sind zum Vergleich die Veränderungen der Koeffizienten mit der in Abschnitt 3.2.1 beschriebenen Helligkeits-/Kontrastnormierung gezeigt.

---

[81] Als Störungen werden hier Bildoperationen bezeichnet, gegenüber denen das eingebettete Wasserzeichen robust sein soll. Dazu zählen in den Simulationen die verlustbehaftete Kompression, additives Rauschen, Tiefpassfilterung sowie Helligkeits- und Kontraständerungen des Trägerbildes.

[82] engl. *probability density function* (*pdf*)

B.1.1 Veränderung der Koeffizienten ohne Helligkeits-/Kontrastnormierung

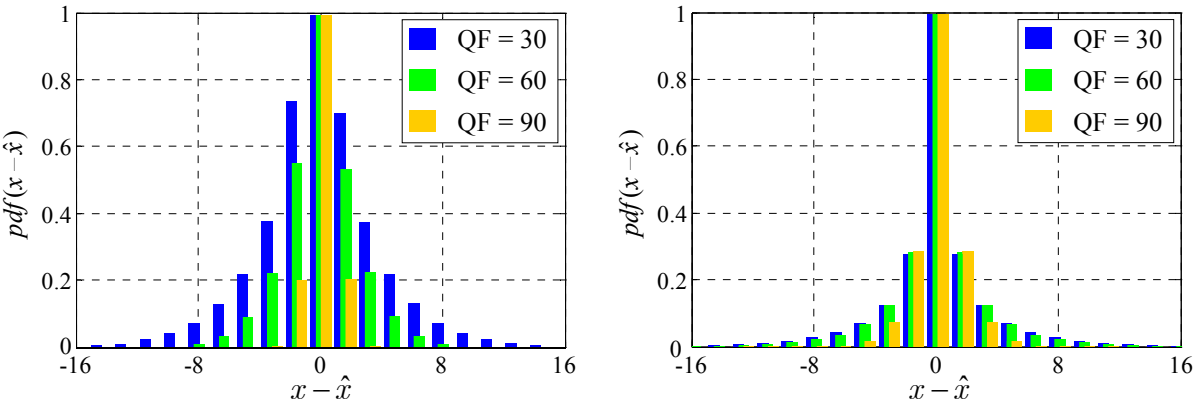


Abbildung B.1 Veränderung der DWT-Koeffizienten des Subbandes LL_1 (links) bzw. der Subbänder HL_1, LH_1 und HH_1 (rechts) nach einer JPEG-Kompression mit unterschiedlichen Stärken QF

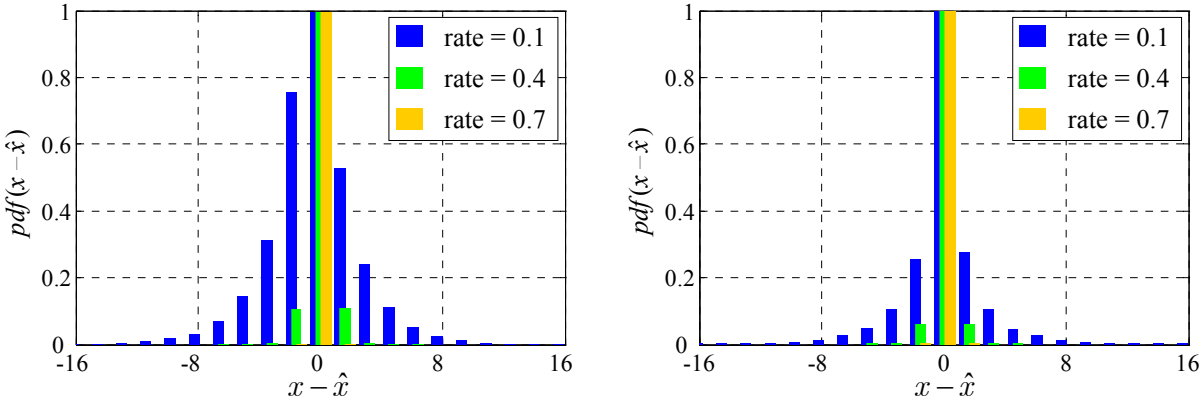


Abbildung B.2 Veränderung der DWT-Koeffizienten des Subbandes LL_1 (links) bzw. der Subbänder HL_1, LH_1 und HH_1 (rechts) nach einer JPEG2000-Kompression mit unterschiedlichen Zielbitraten *bpp*

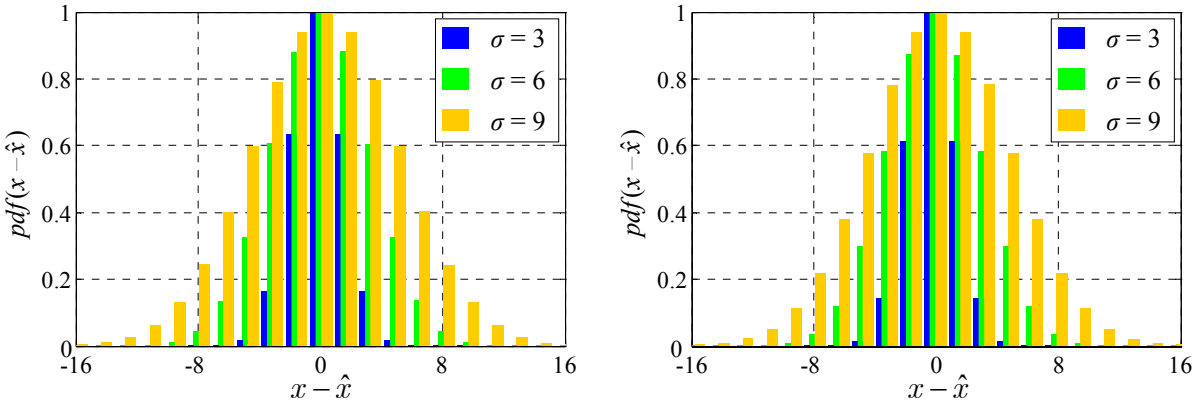


Abbildung B.3 Veränderung der DWT-Koeffizienten des Subbandes LL_1 (links) bzw. der Subbänder HL_1, LH_1 und HH_1 (rechts) nach additivem weißen Gaußschen Rauschen mit unterschiedlichen Varianzen σ

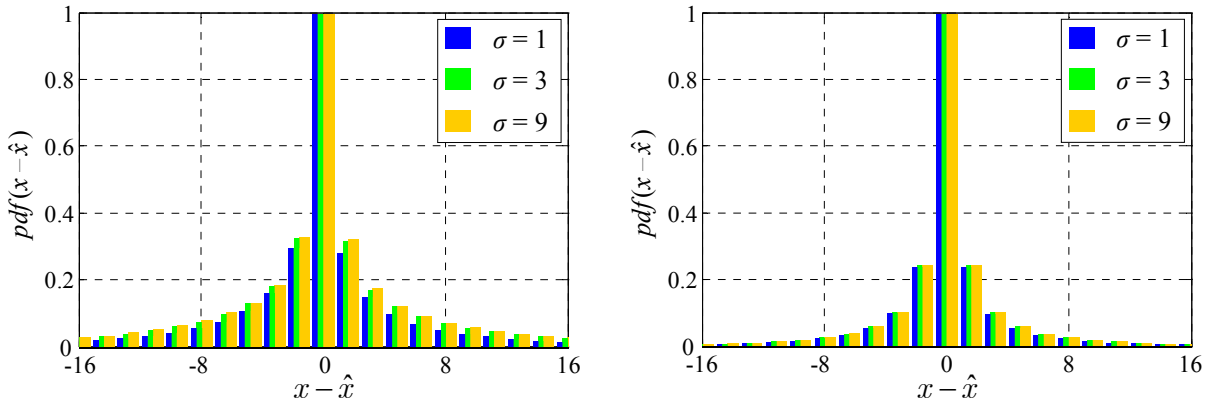


Abbildung B.4 Veränderung der DWT-Koeffizienten des Subbandes LL_1 (links) bzw. der Subbänder HL_1, LH_1 und HH_1 (rechts) nach Gaußscher Tiefpassfilterung mit unterschiedlichen Varianzen σ

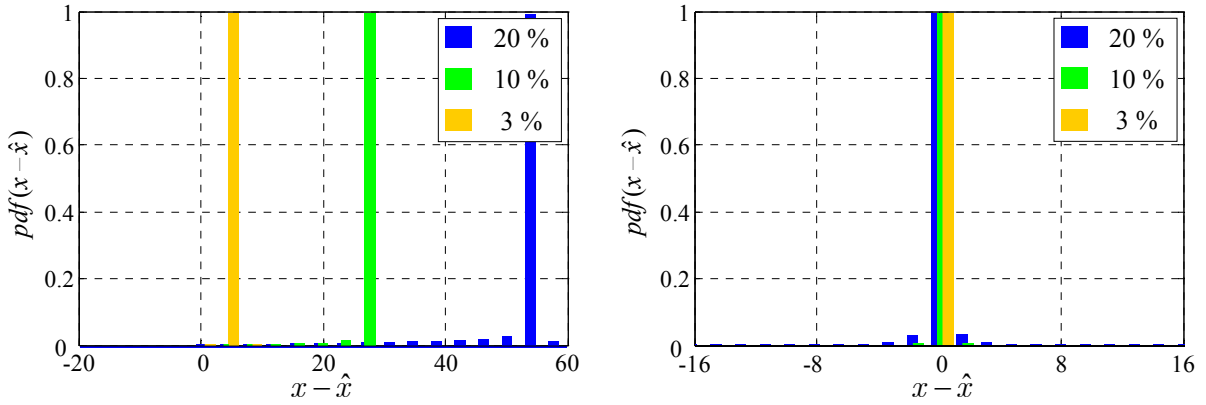


Abbildung B.5 Veränderung der DWT-Koeffizienten des Subbandes LL_1 (links) bzw. der Subbänder HL_1, LH_1 und HH_1 (rechts) nach prozentualer Reduzierung der Bildhelligkeit, bezogen auf den Grauwertebereich $[0;255]$

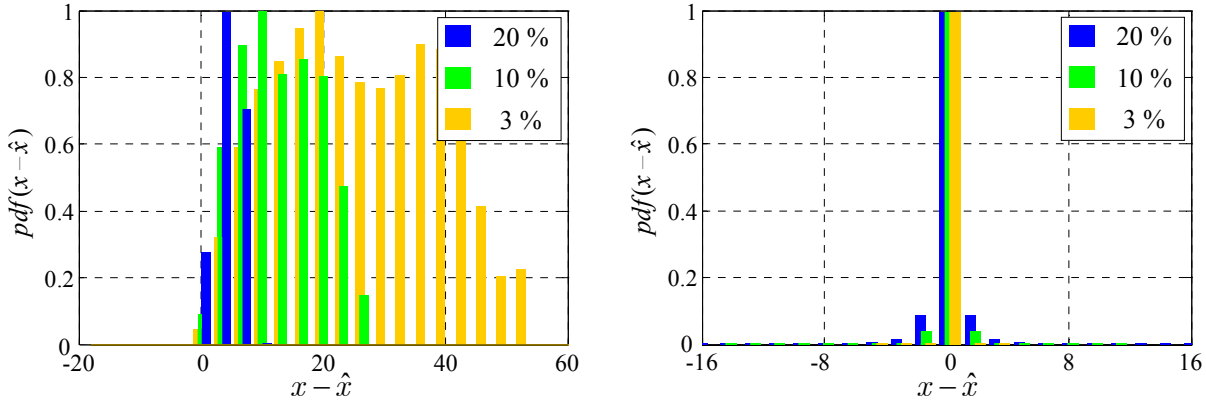


Abbildung B.6 Veränderung der DWT-Koeffizienten des Subbandes LL_1 (links) bzw. der Subbänder HL_1, LH_1 und HH_1 (rechts) nach prozentualer Reduzierung des Bildkontrastes, bezogen auf den Grauwertebereich $[0;255]$

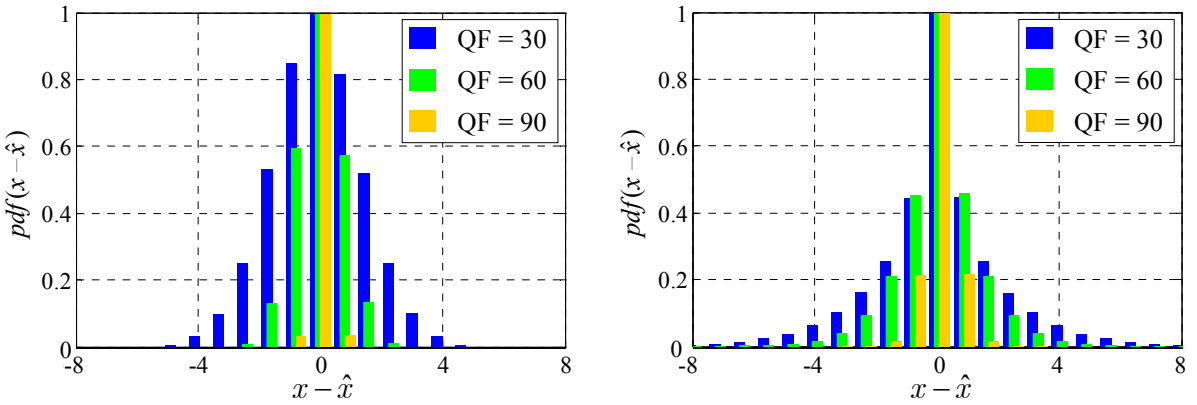


Abbildung B.7 Veränderung der DWT-Koeffizienten des Subbandes LL_2 (links) bzw. der Subbänder HL_2, LH_2 und HH_2 (rechts) nach einer JPEG-Kompression mit unterschiedlichen Stärken QF

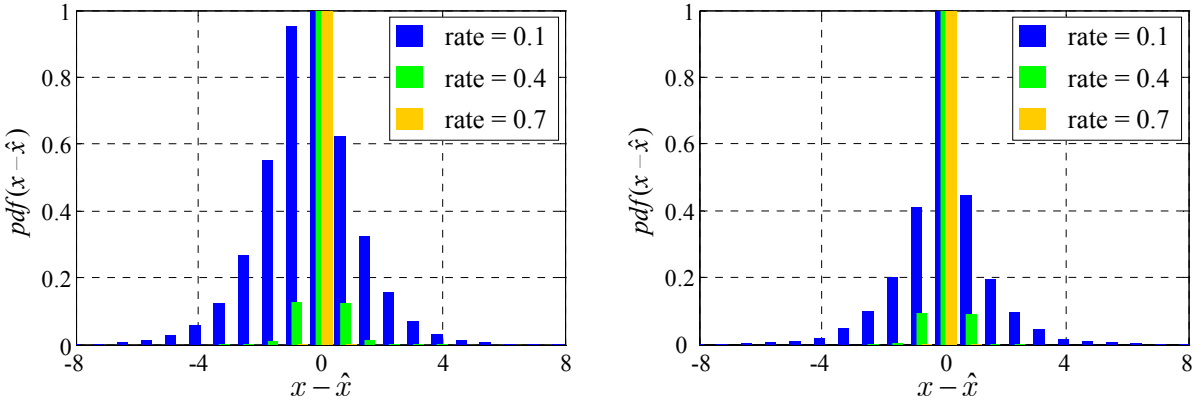


Abbildung B.8 Veränderung der DWT-Koeffizienten des Subbandes LL_2 (links) bzw. der Subbänder HL_2, LH_2 und HH_2 (rechts) nach einer JPEG2000-Kompression mit unterschiedlichen Zielbitraten *bpp*

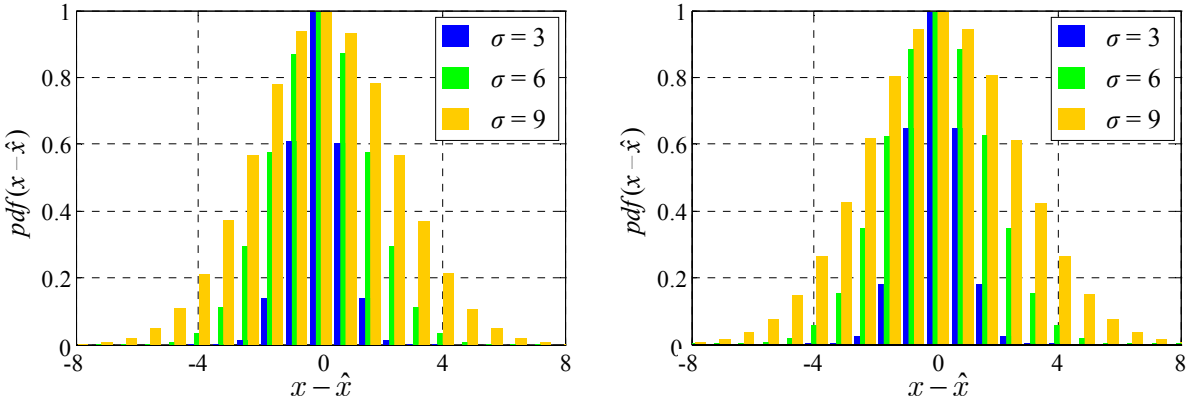


Abbildung B.9 Veränderung der DWT-Koeffizienten des Subbandes LL_2 (links) bzw. der Subbänder HL_2, LH_2 und HH_2 (rechts) nach additivem weißen Gaußschen Rauschen mit unterschiedlichen Varianzen σ

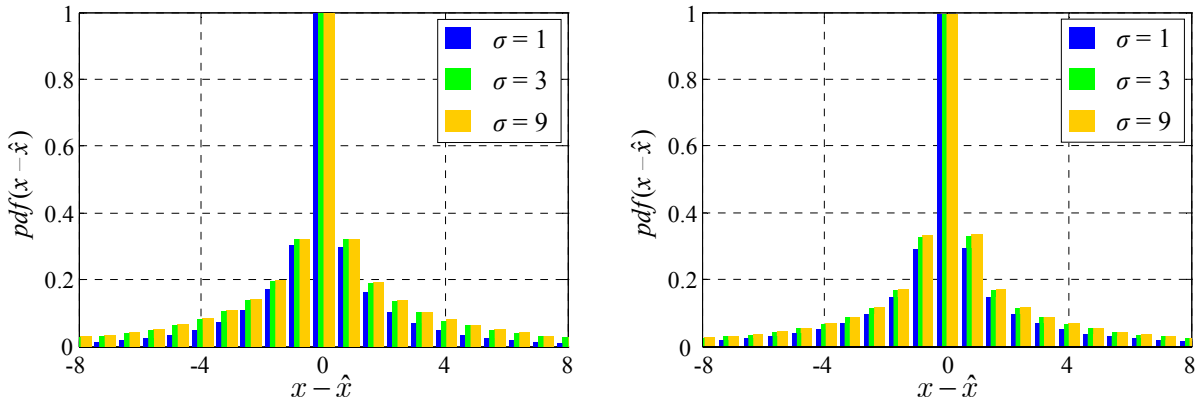


Abbildung B.10 Veränderung der DWT-Koeffizienten des Subbandes LL_2 (links) bzw. der Subbänder HL_2, LH_2 und HH_2 (rechts) nach Gaußscher Tiefpassfilterung mit unterschiedlichen Varianzen σ

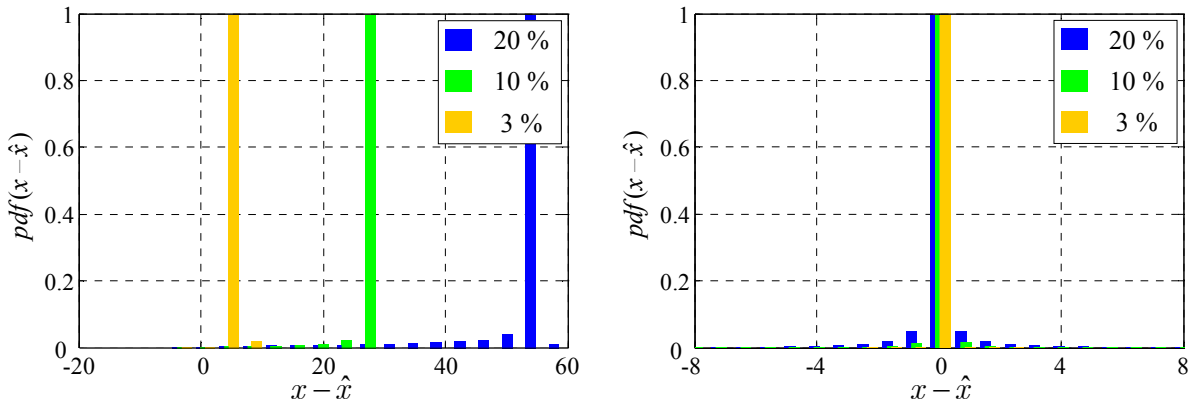


Abbildung B.11 Veränderung der DWT-Koeffizienten des Subbandes LL_2 (links) bzw. der Subbänder HL_2, LH_2 und HH_2 (rechts) nach prozentualer Reduzierung der Bildhelligkeit, bezogen auf den Grauwertebereich $[0;255]$

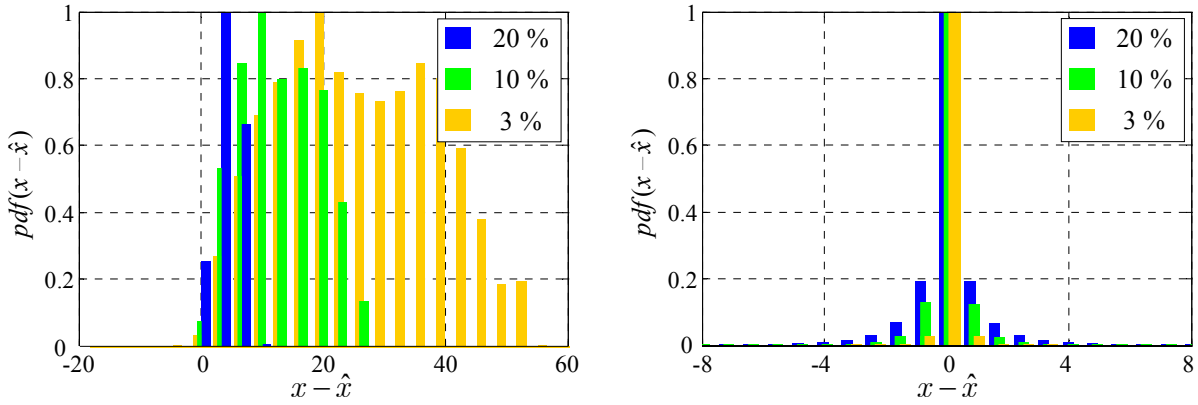


Abbildung B.12 Veränderung der DWT-Koeffizienten des Subbandes LL_2 (links) bzw. der Subbänder HL_2, LH_2 und HH_2 (rechts) nach prozentualer Reduzierung des Bildkontrastes, bezogen auf den Grauwertebereich $[0;255]$

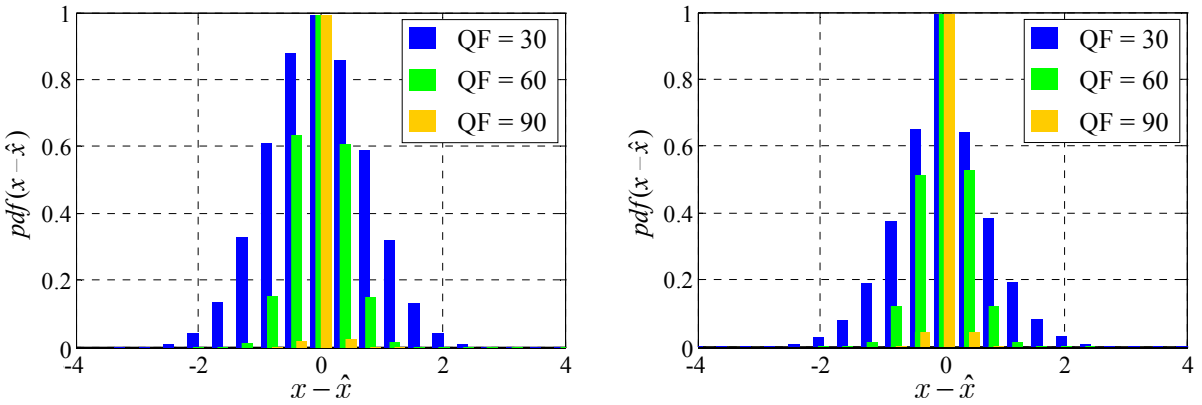


Abbildung B.13 Veränderung der DWT-Koeffizienten des Subbandes LL_3 (links) bzw. der Subbänder HL_3, LH_3 und HH_3 (rechts) nach einer JPEG-Kompression mit unterschiedlichen Stärken QF

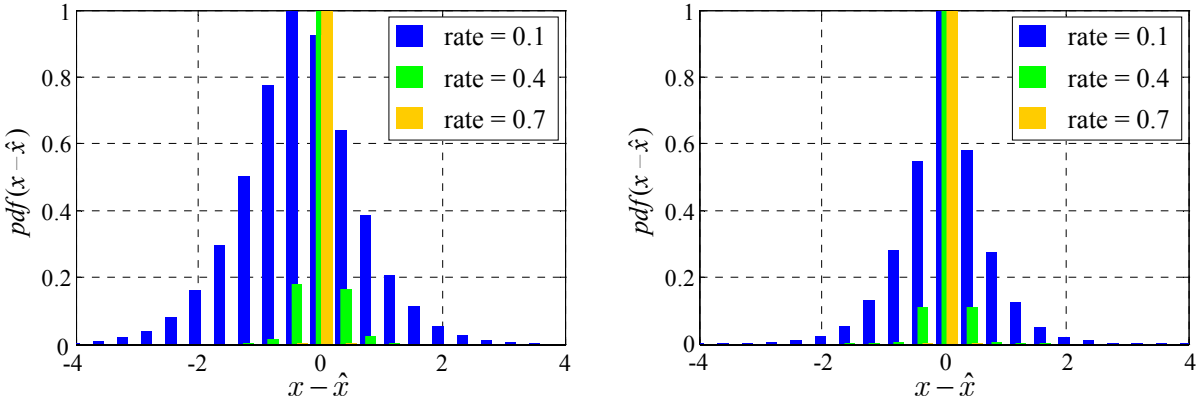


Abbildung B.14 Veränderung der DWT-Koeffizienten des Subbandes LL_3 (links) bzw. der Subbänder HL_3, LH_3 und HH_3 (rechts) nach einer JPEG2000-Kompression mit unterschiedlichen Zielbitraten *bpp*

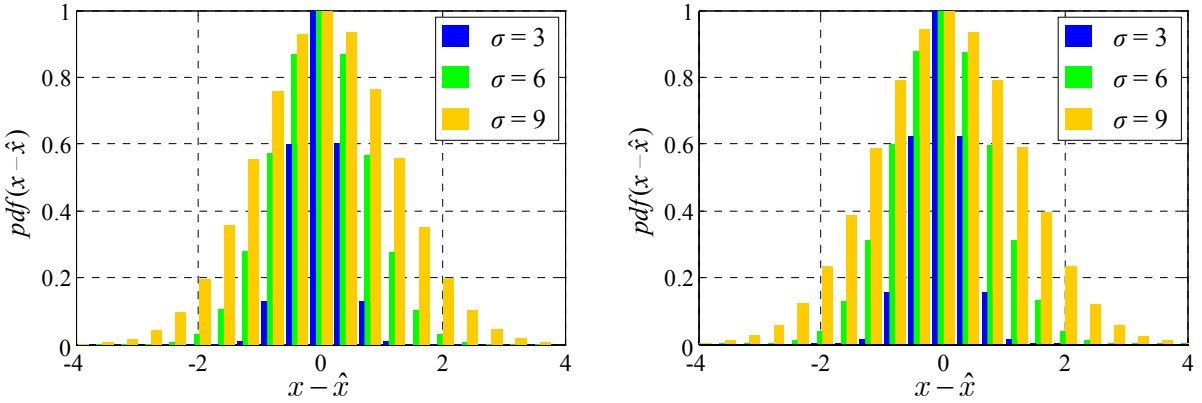


Abbildung B.15 Veränderung der DWT-Koeffizienten des Subbandes LL_3 (links) bzw. der Subbänder HL_3, LH_3 und HH_3 (rechts) nach additivem weißen Gaußschen Rauschen mit unterschiedlichen Varianzen σ

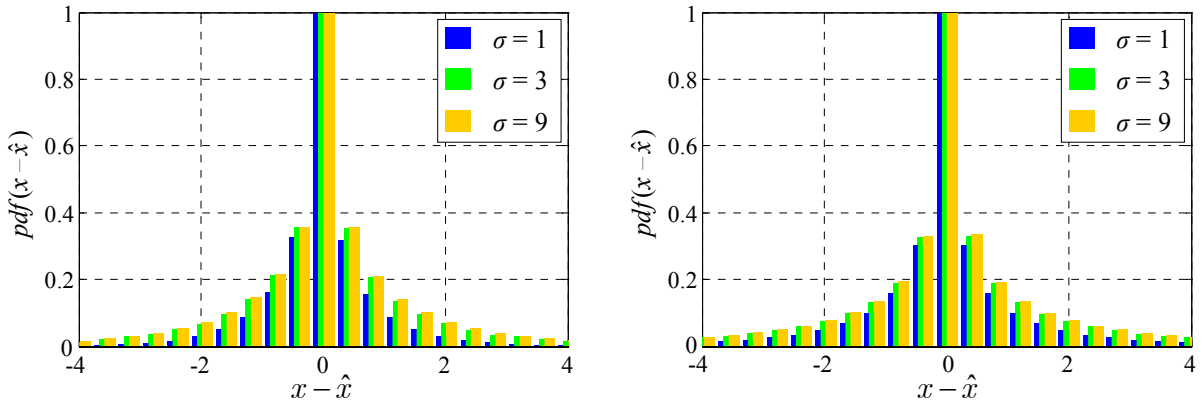


Abbildung B.16 Veränderung der DWT-Koeffizienten des Subbandes LL_3 (links) bzw. der Subbänder HL_3, LH_3 und HH_3 (rechts) nach Gaußscher Tiefpassfilterung mit unterschiedlichen Varianzen σ

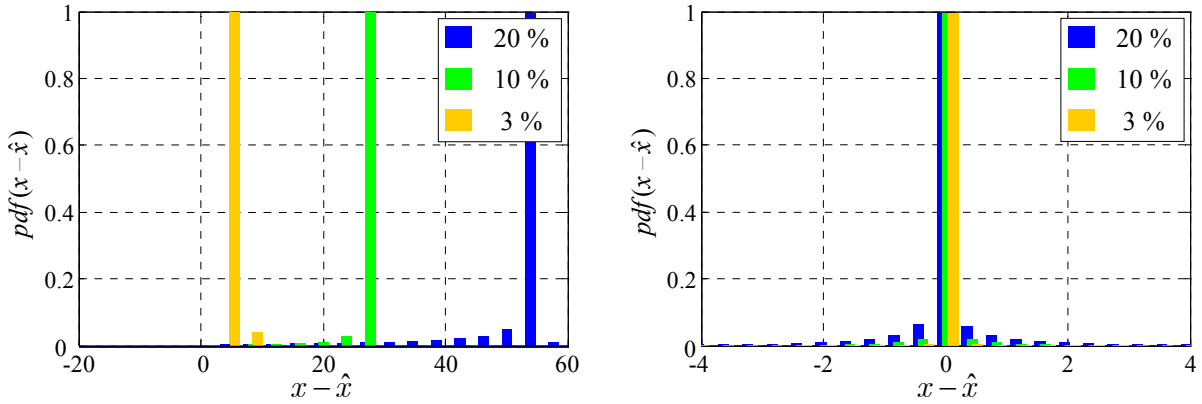


Abbildung B.17 Veränderung der DWT-Koeffizienten des Subbandes LL_3 (links) bzw. der Subbänder HL_3, LH_3 und HH_3 (rechts) nach prozentualer Reduzierung der Bildhelligkeit, bezogen auf den Grauwertebereich $[0;255]$

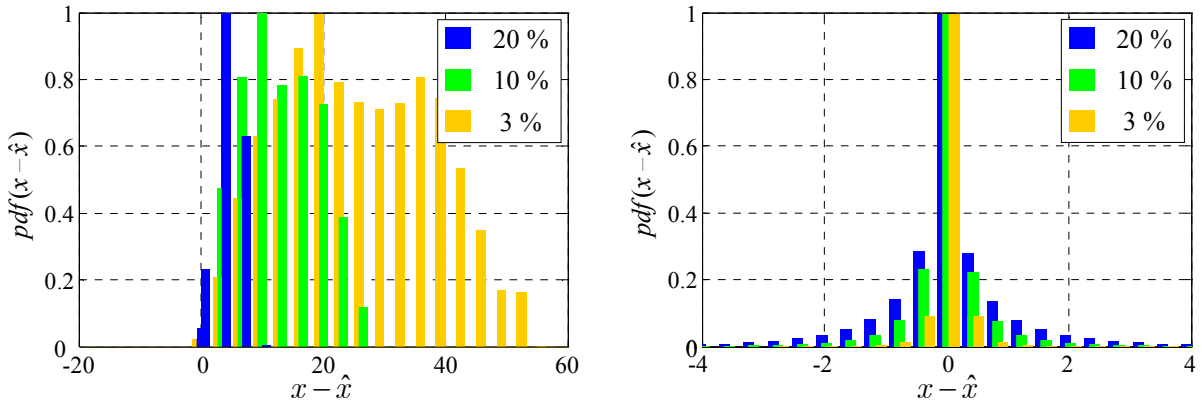


Abbildung B.18 Veränderung der DWT-Koeffizienten des Subbandes LL_3 (links) bzw. der Subbänder HL_3, LH_3 und HH_3 (rechts) nach prozentualer Reduzierung des Bildkontrastes, bezogen auf den Grauwertebereich $[0;255]$

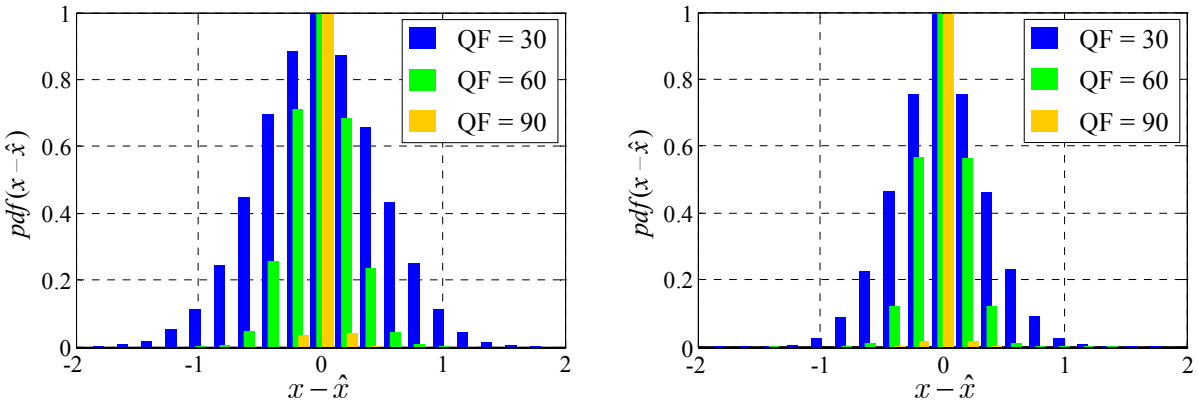


Abbildung B.19 Veränderung der DWT-Koeffizienten des Subbandes LL_4 (links) bzw. der Subbänder HL_4, LH_4 und HH_4 (rechts) nach einer JPEG-Kompression mit unterschiedlichen Stärken QF

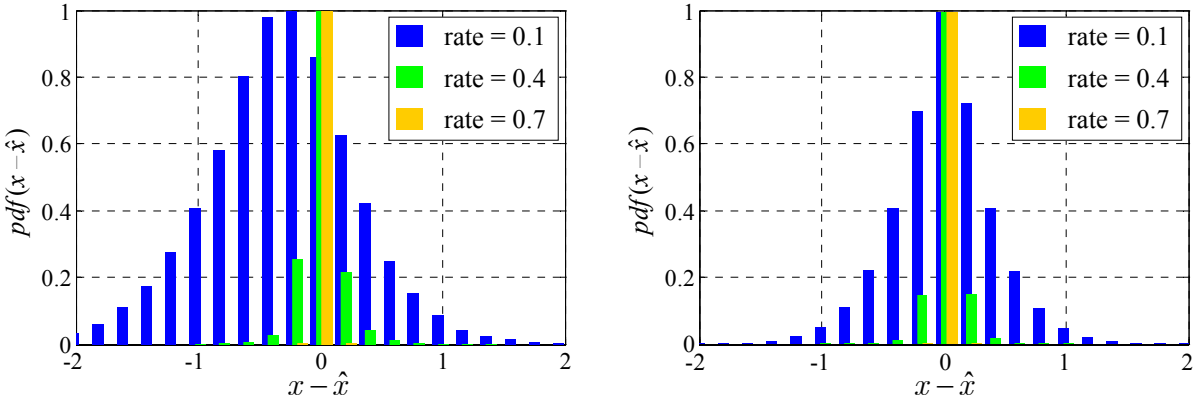


Abbildung B.20 Veränderung der DWT-Koeffizienten des Subbandes LL_4 (links) bzw. der Subbänder HL_4, LH_4 und HH_4 (rechts) nach einer JPEG2000-Kompression mit unterschiedlichen Zielbitraten *bpp*

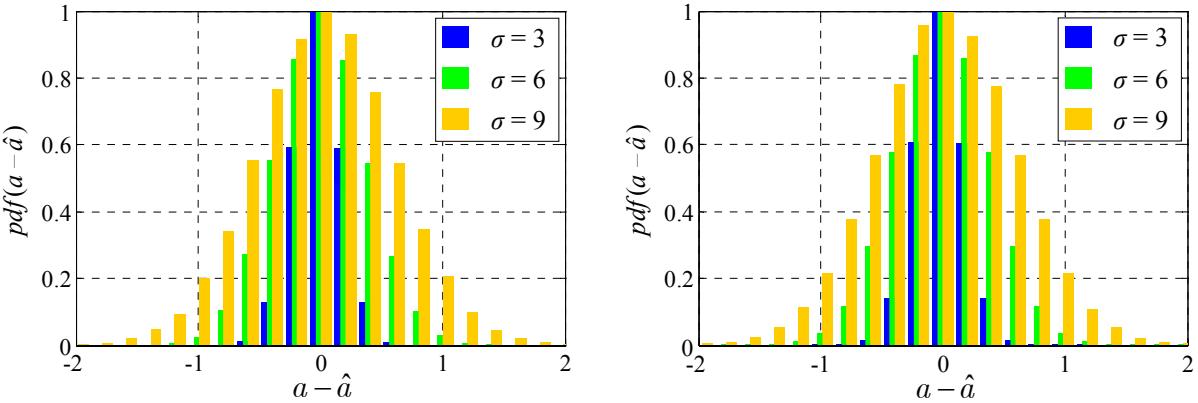


Abbildung B.21 Veränderung der DWT-Koeffizienten des Subbandes LL_4 (links) bzw. der Subbänder HL_4, LH_4 und HH_4 (rechts) nach additivem weißen Gaußschen Rauschen mit unterschiedlichen Varianzen σ

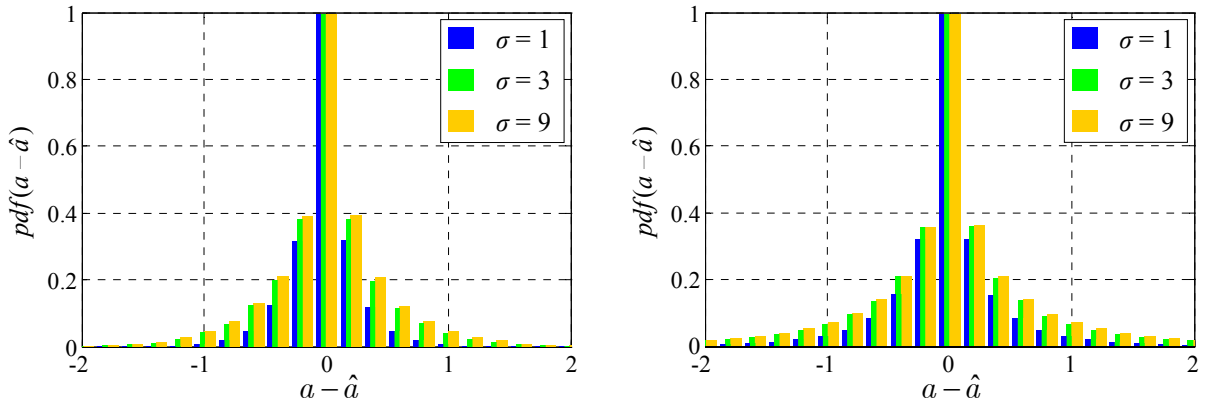


Abbildung B.22 Veränderung der DWT-Koeffizienten des Subbandes LL_4 (links) bzw. der Subbänder HL_4, LH_4 und HH_4 (rechts) nach Gaußscher Tiefpassfilterung mit unterschiedlichen Varianzen σ

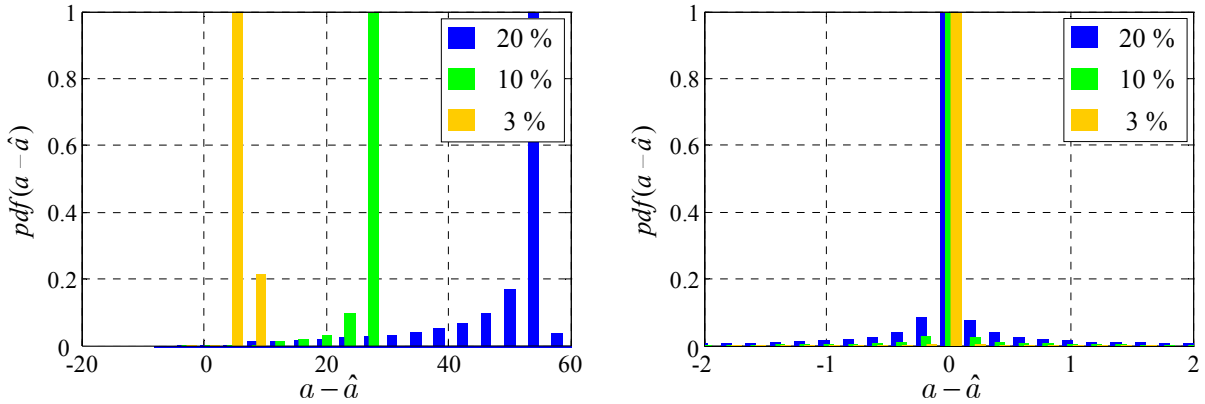


Abbildung B.23 Veränderung der DWT-Koeffizienten des Subbandes LL_4 (links) bzw. der Subbänder HL_4, LH_4 und HH_4 (rechts) nach prozentualer Reduzierung der Bildhelligkeit, bezogen auf den Grauwertebereich $[0;255]$

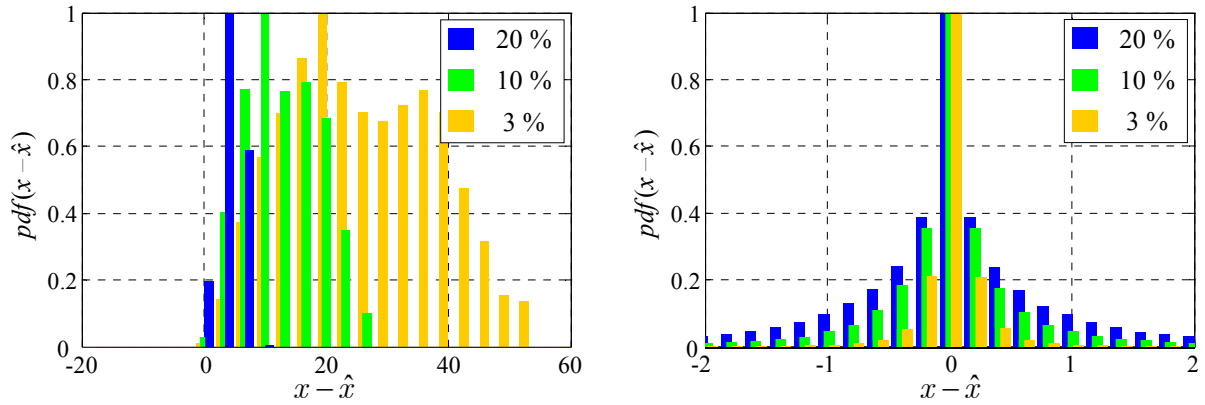


Abbildung B.24 Veränderung der DWT-Koeffizienten des Subbandes LL_4 (links) bzw. der Subbänder HL_4, LH_4 und HH_4 (rechts) nach prozentualer Reduzierung des Bildkontrastes, bezogen auf den Grauwertebereich $[0;255]$

B.1.2 Veränderung der Koeffizienten mit Helligkeits-/Kontrastnormierung

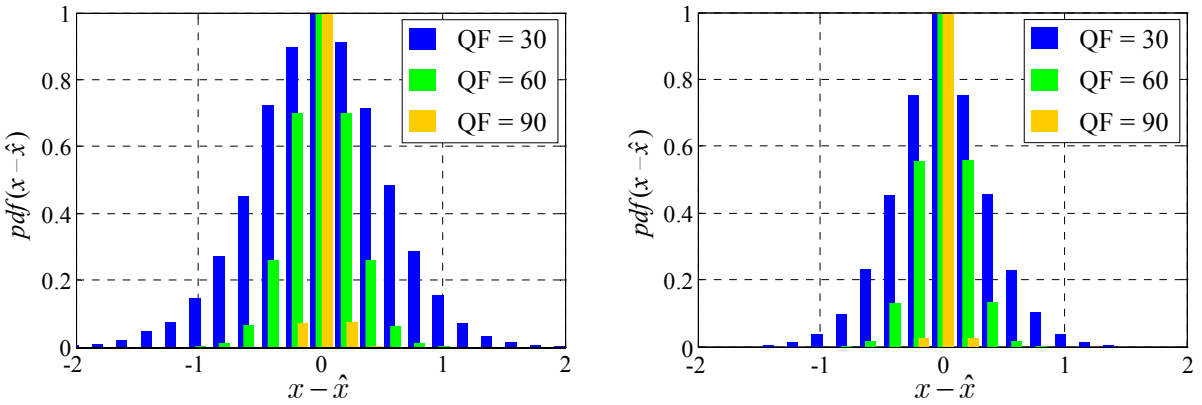


Abbildung B.25 Veränderung der DWT-Koeffizienten des Subbandes LL_4 (links) bzw. der Subbänder HL_4, LH_4 und HH_4 (rechts) nach einer JPEG-Kompression mit unterschiedlichen Stärken QF

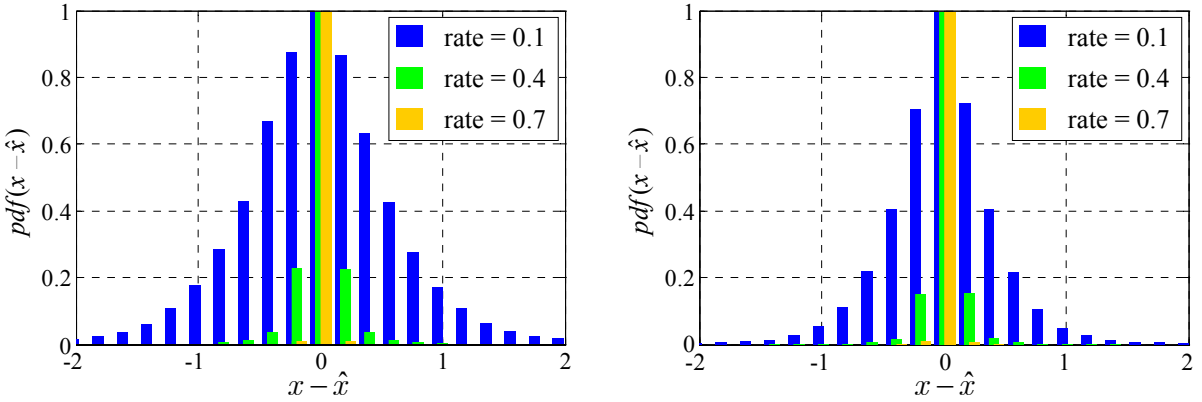


Abbildung B.26 Veränderung der DWT-Koeffizienten des Subbandes LL_4 (links) bzw. der Subbänder HL_4, LH_4 und HH_4 (rechts) nach einer JPEG2000-Kompression mit unterschiedlichen Zielbitraten *bpp*

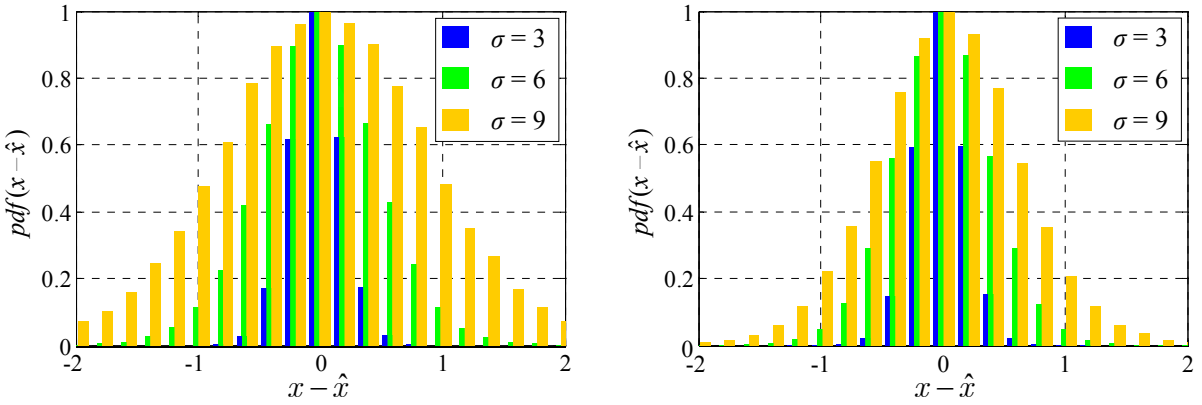


Abbildung B.27 Veränderung der DWT-Koeffizienten des Subbandes LL_4 (links) bzw. der Subbänder HL_4, LH_4 und HH_4 (rechts) nach additivem weißen Gaußschen Rauschen mit unterschiedlichen Varianzen σ

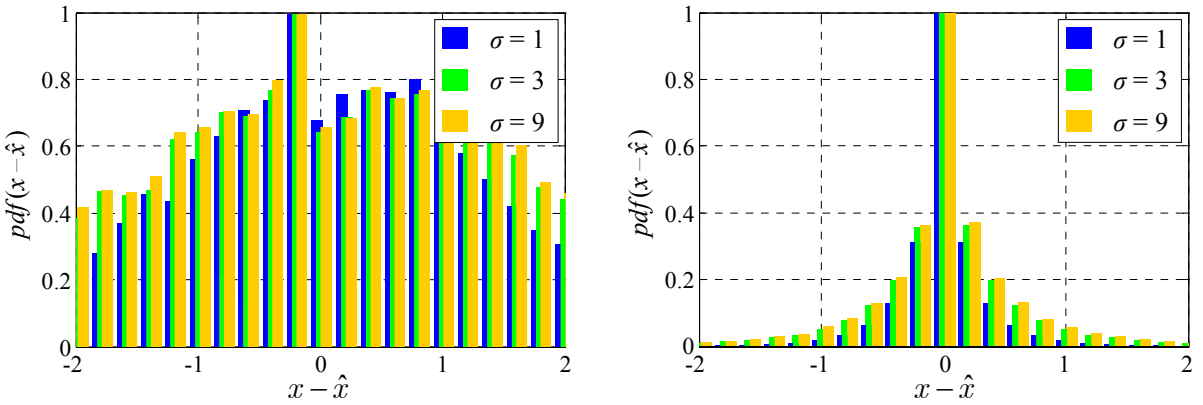


Abbildung B.28 Veränderung der DWT-Koeffizienten des Subbandes LL_4 (links) bzw. der Subbänder HL_4, LH_4 und HH_4 (rechts) nach Gaußscher Tiefpassfilterung mit unterschiedlichen Varianzen σ

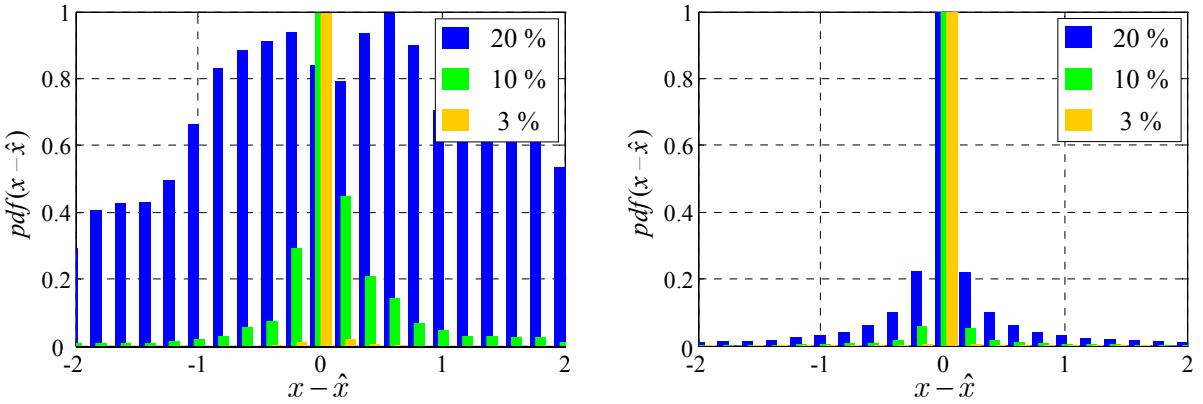


Abbildung B.29 Veränderung der DWT-Koeffizienten des Subbandes LL_4 (links) bzw. der Subbänder HL_4, LH_4 und HH_4 (rechts) nach prozentualer Reduzierung der Bildhelligkeit, bezogen auf den Grauwertebereich $[0;255]$

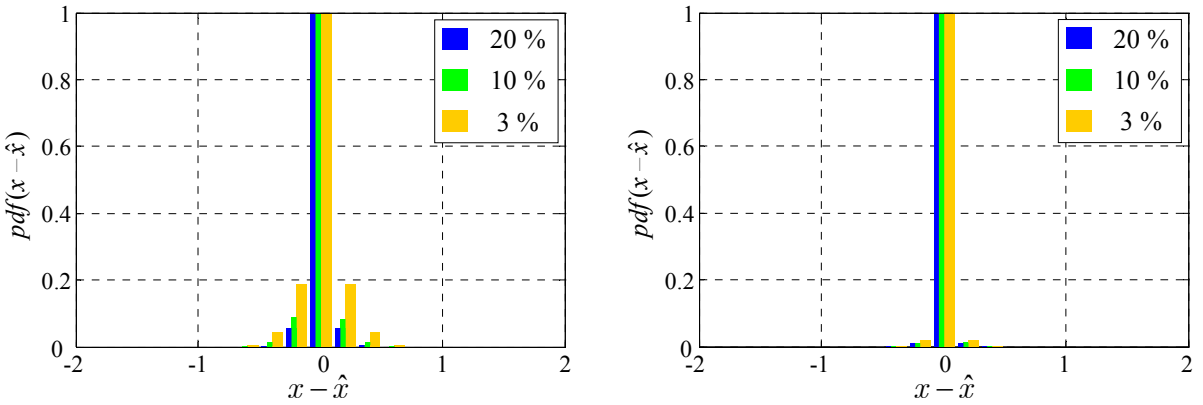


Abbildung B.30 Veränderung der DWT-Koeffizienten des Subbandes LL_4 (links) bzw. der Subbänder HL_4, LH_4 und HH_4 (rechts) nach prozentualer Reduzierung des Bildkontrastes, bezogen auf den Grauwertebereich $[0;255]$

B.2 Bildverzerrungen durch die Quantisierung der Koeffizienten

Die beiden nachfolgenden Abbildungen zeigen die anhand von Simulationen berechneten, über 52 Bilder gemittelten Bildverzerrungen, die durch die Quantisierung der DWT-Koeffizienten entstehen. Für die zu den Kurven in Abbildung B.31 (links) führenden Berechnungen wurde die in den Gleichungen (2.4) und (2.5) beschriebene "Quantisierung mit Totzone" (Abschnitt 2.2) verwendet, die auch im JPEG2000-Kompressionsprozess zum Einsatz kommt. Werte innerhalb der Totzone $[-\Delta;+\Delta]$ werden dabei zu Null quantisiert/rekonstruiert. Da jedoch viele der DWT-Koeffizienten in den HL-, LH- und HH-Subbändern sehr kleine Werte besitzen, führt diese Quantisierung zu starken Bildverzerrungen. Abbildung B.31 (rechts) zeigt hingegen die Bildverzerrungen für die in Abschnitt 3.2.1 neu entwickelte Quantisierung.

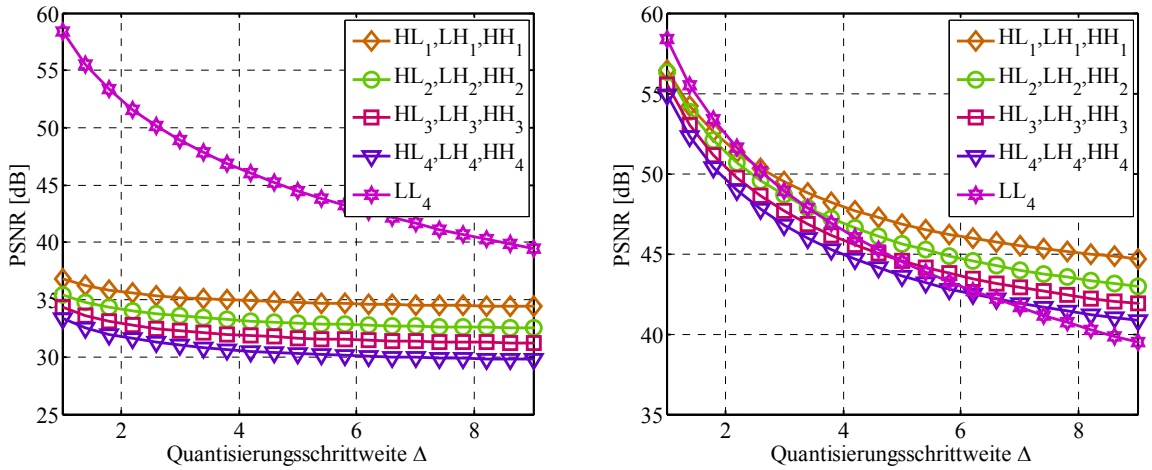


Abbildung B.31 Bildverzerrungen (PSNR) durch die Quantisierung der DWT-Koeffizienten unterschiedlicher Subbänder mit der Schrittweite Δ. (links) Quantisierung von Koeffizienten in der Totzone zu Null. (rechts) keine Quantisierung von Koeffizienten mit Werten im Bereich $[-\Delta/2,+\Delta/2]$

B.3 Robustheit der Koeffizienten gegenüber Störungen

In einer weiteren Simulation wurde die Robustheit der quantisierten Koeffizienten gegenüber Störungen untersucht. Dazu wurden jeweils die DWT-Koeffizienten der 52 verwendeten Testbilder berechnet, mit unterschiedlichen Schrittweiten $\Delta = 1$ bis $\Delta = 5$ quantisiert und zurück in den Pixelbereich transformiert. Nach der Anwendung von Störungen mit unterschiedlichen Parametern wurden die DWT-Zerlegungen ein zweites Mal durchgeführt. Quantisierungen der gestörten Koeffizienten zu anderen als den ursprünglichen Quantisierungsintervallen wurden als Fehler gezählt. Die anhand aller Testbilder ermittelte Fehlerauftretenswahrscheinlichkeit ist in den nachfolgenden Abbildungen dargestellt. Dabei wird zwischen den LL-Band- und den HL-, LH-, HH-Band-Koeffizienten unterschieden. Letztere sind gemeinsam abgebildet, da sie ähnliche Fehlerkurven besitzen, die sich vor allem bei einer Helligkeits- oder Kontraständerung des Trägerbildes von denen der LL-Band-Koeffizienten unterscheiden.

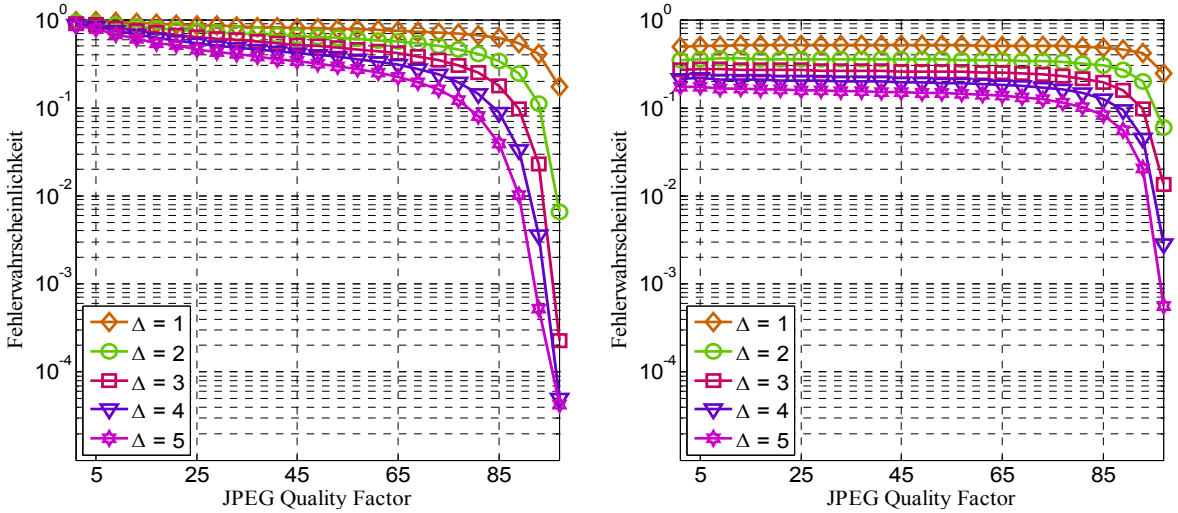


Abbildung B.32 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_1 (links) bzw. HL_1, LH_1, HH_1 (rechts) gegenüber JPEG-Kompression

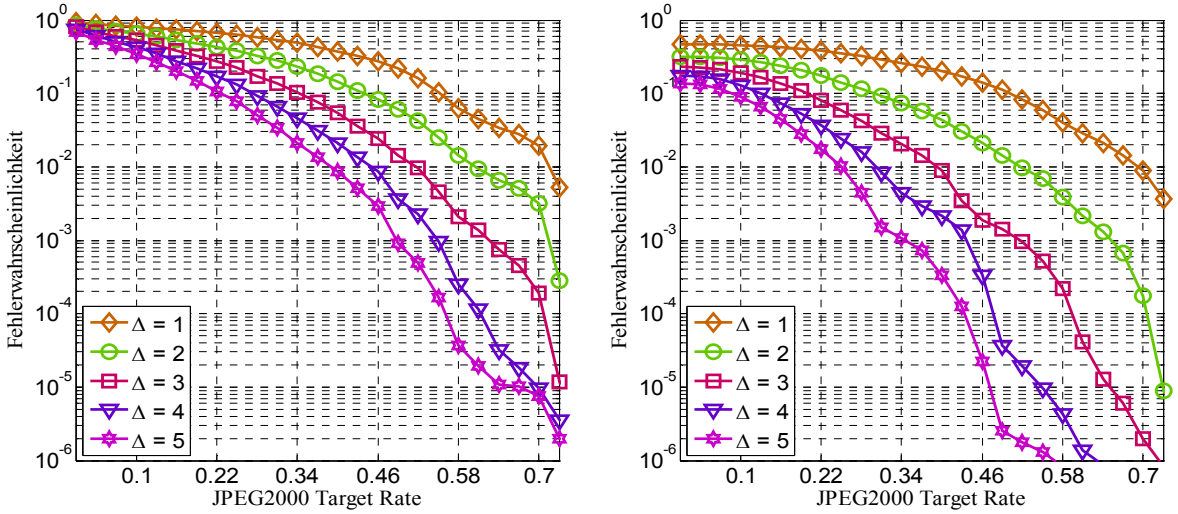


Abbildung B.33 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_1 (links) bzw. HL_1, LH_1, HH_1 (rechts) gegenüber JPEG2000-Kompression

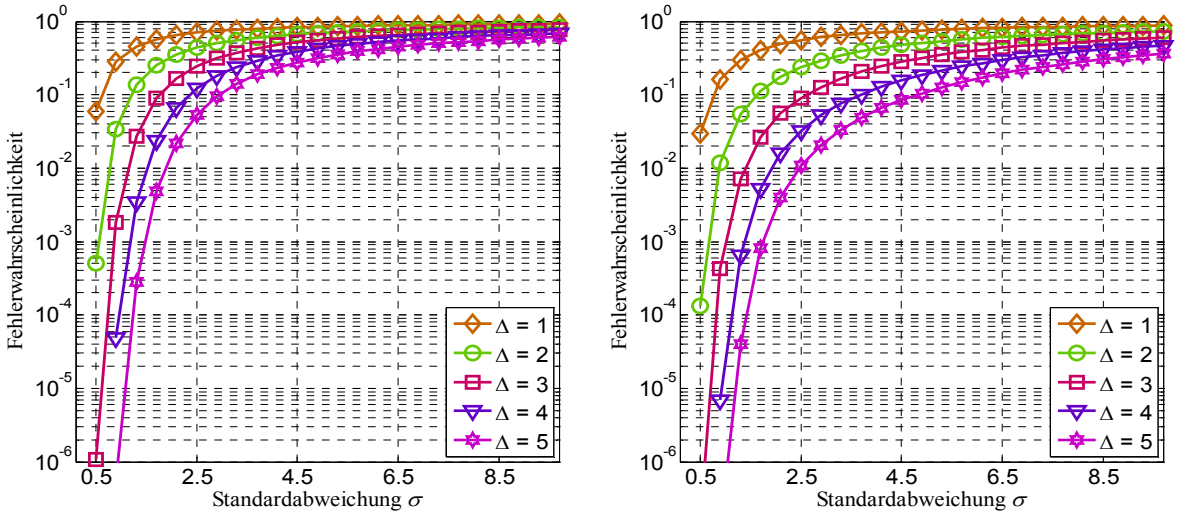


Abbildung B.34 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_1 (links) bzw. HL_1, LH_1, HH_1 (rechts) gegenüber Gaußschem Rauschen

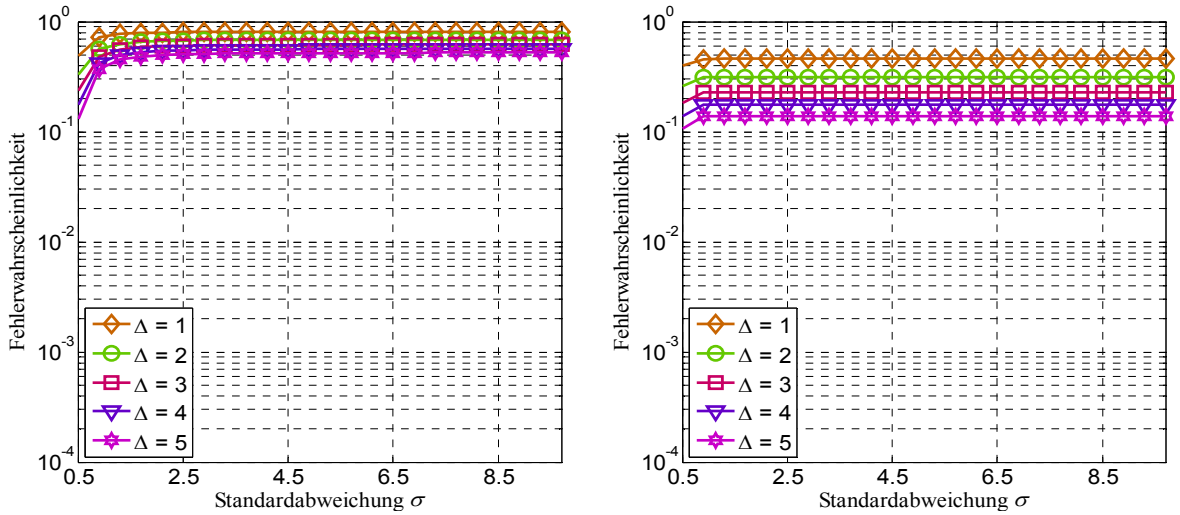


Abbildung B.35 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_1 (links) bzw. HL_1, LH_1, HH_1 (rechts) gegenüber Gaußscher Tiefpassfilterung

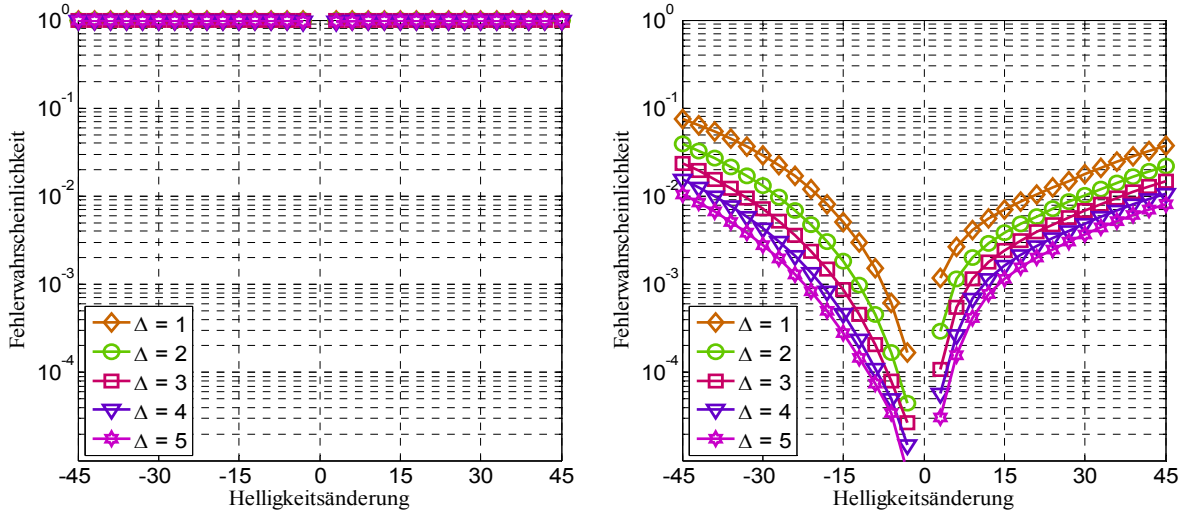


Abbildung B.36 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_1 (links) bzw. HL_1, LH_1, HH_1 (rechts) gegenüber Helligkeitsänderungen bezogen auf den Grauwertebereich $[0;255]$

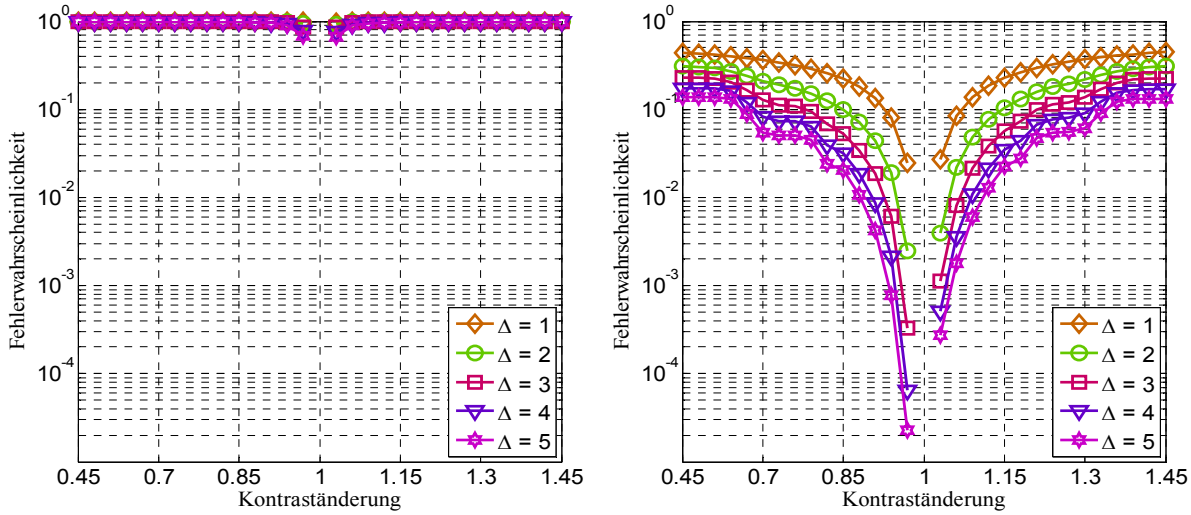


Abbildung B.37 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_1 (links) bzw. HL_1, LH_1, HH_1 (rechts) gegenüber Kontraständerungen bezogen auf den Grauwertebereich $[0;255]$

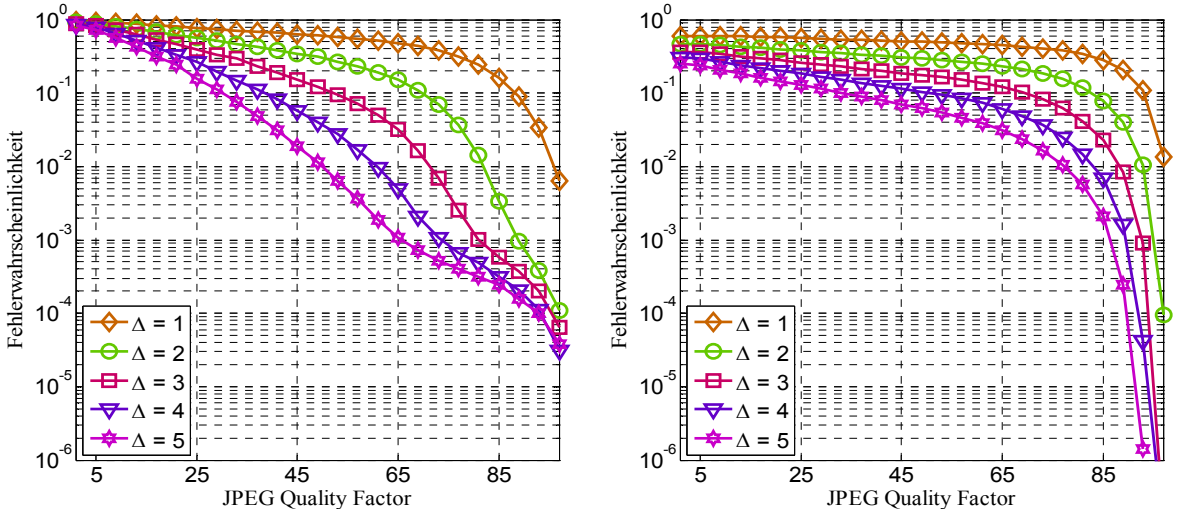


Abbildung B.38 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_2 (links) bzw. HL_2, LH_2, HH_2 (rechts) gegenüber JPEG-Kompression

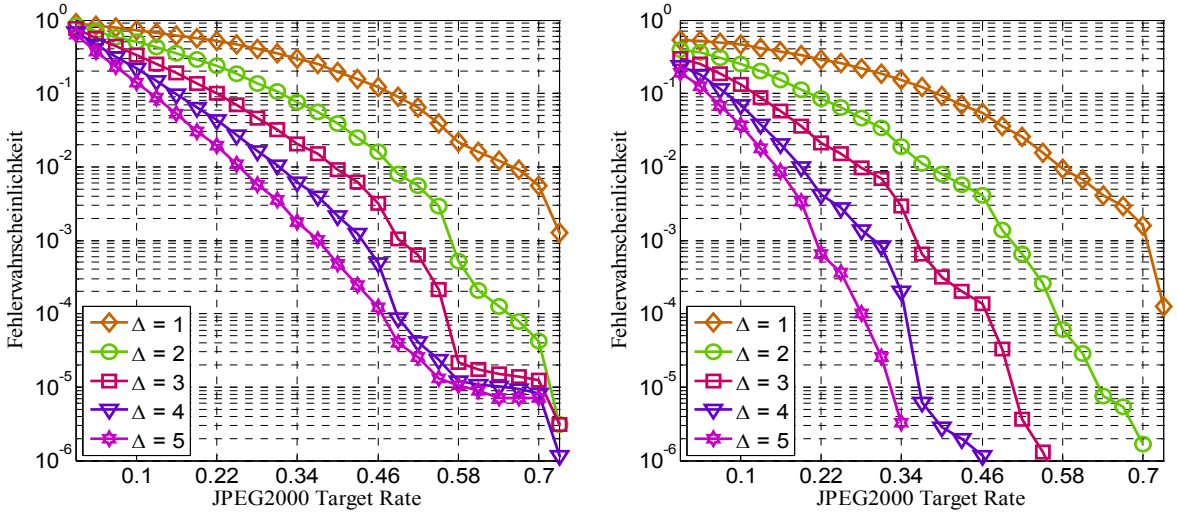


Abbildung B.39 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_2 (links) bzw. HL_2, LH_2, HH_2 (rechts) gegenüber JPEG2000-Kompression

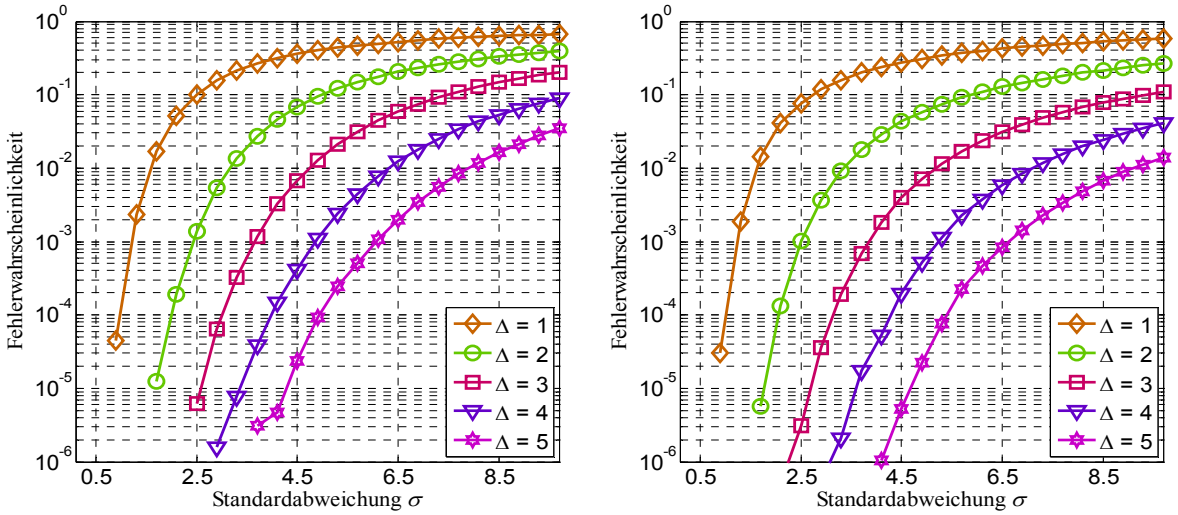


Abbildung B.40 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_2 (links) bzw. HL_2, LH_2, HH_2 (rechts) gegenüber Gaußschem Rauschen

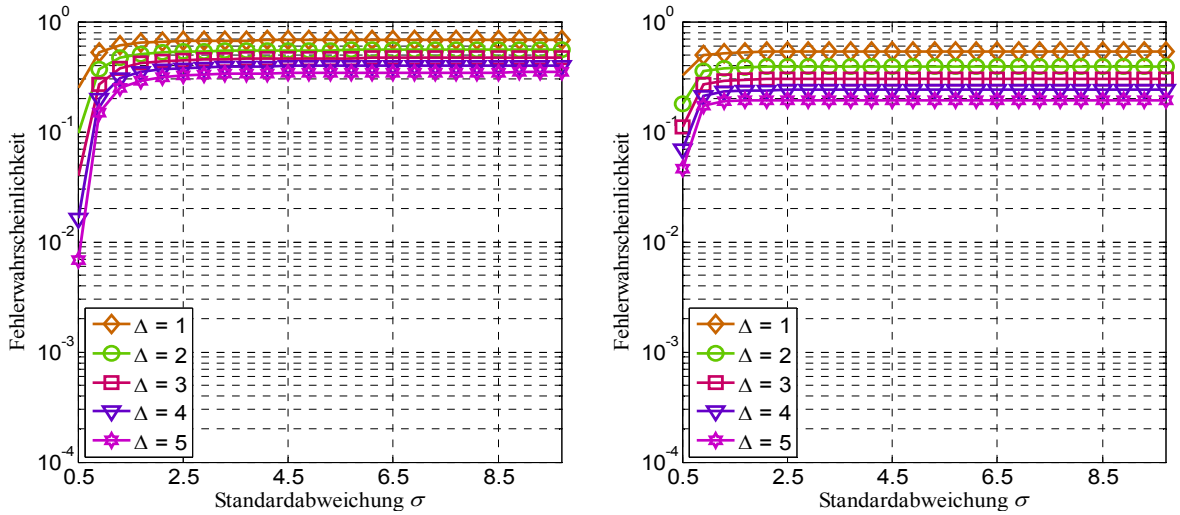


Abbildung B.41 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_2 (links) bzw. HL_2, LH_2, HH_2 (rechts) gegenüber Gaußscher Tiefpassfilterung

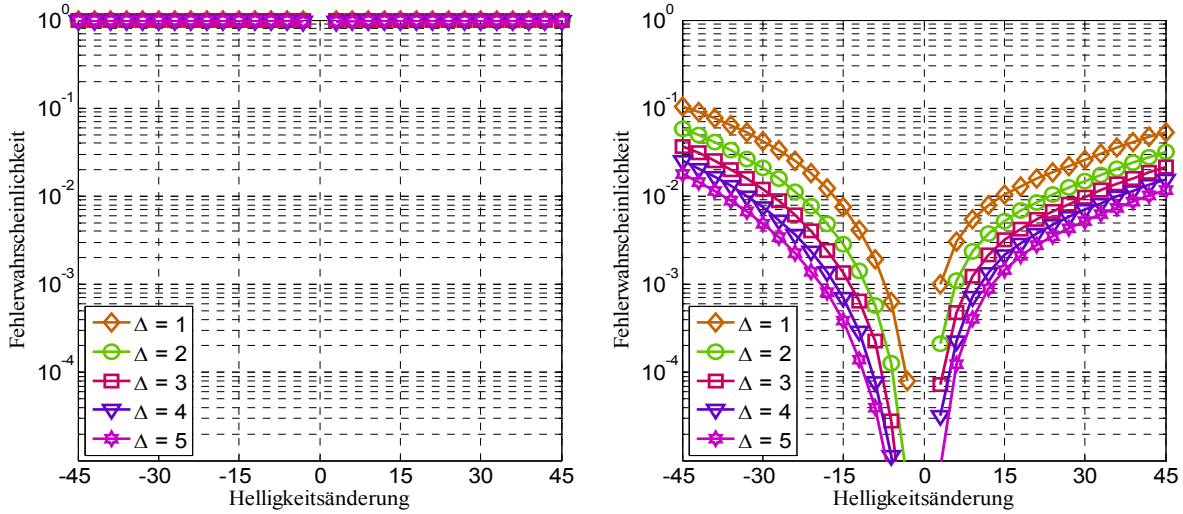


Abbildung B.42 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_2 (links) bzw. HL_2, LH_2, HH_2 (rechts) gegenüber Helligkeitsänderungen bezogen auf den Grauwertebereich $[0;255]$

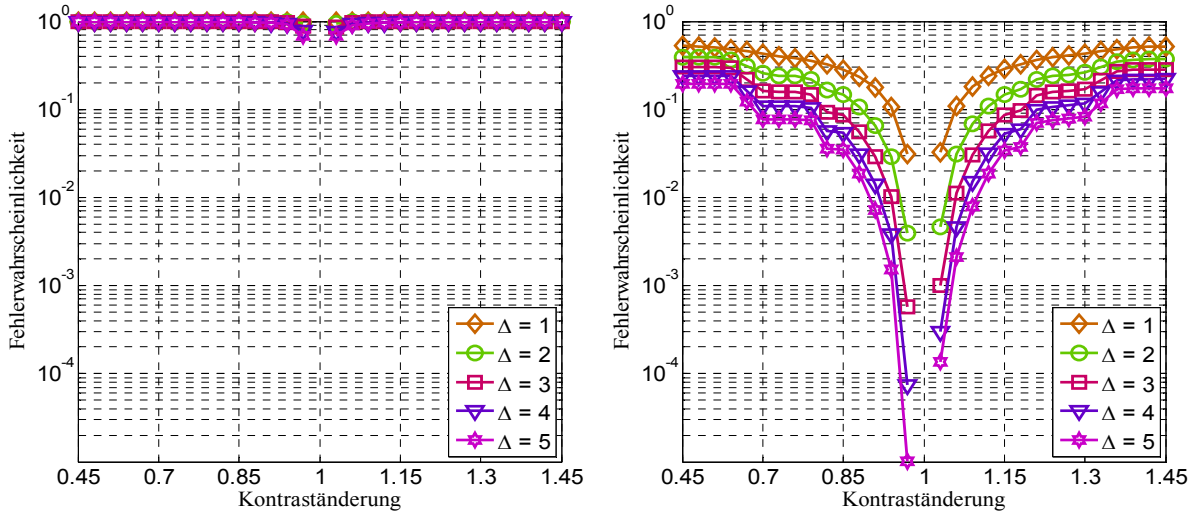


Abbildung B.43 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_2 (links) bzw. HL_2, LH_2, HH_2 (rechts) gegenüber Kontraständerungen bezogen auf den Grauwertebereich $[0;255]$

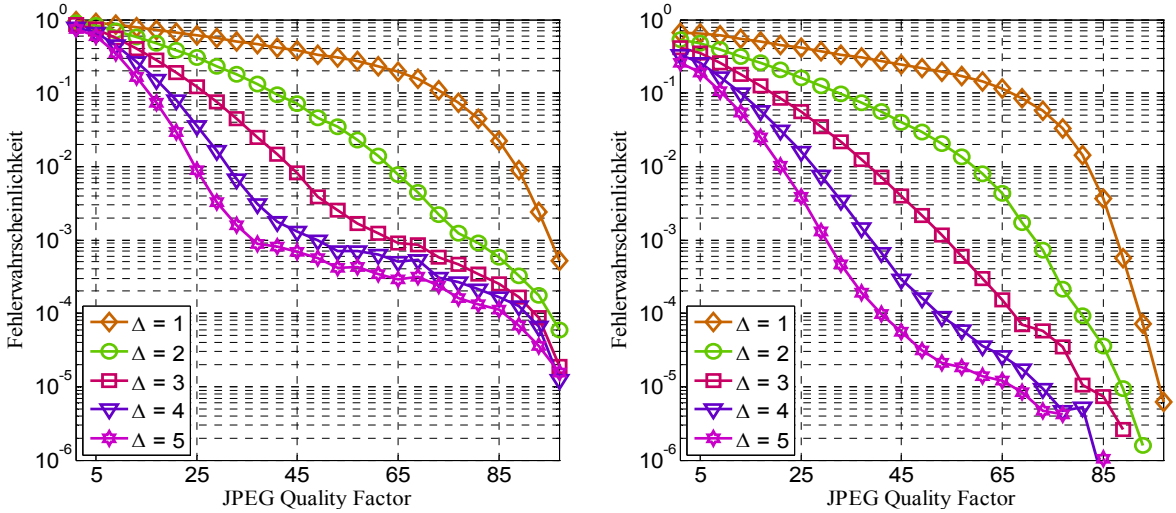


Abbildung B.44 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_3 (links) bzw. HL_3, LH_3, HH_3 (rechts) gegenüber JPEG-Kompression

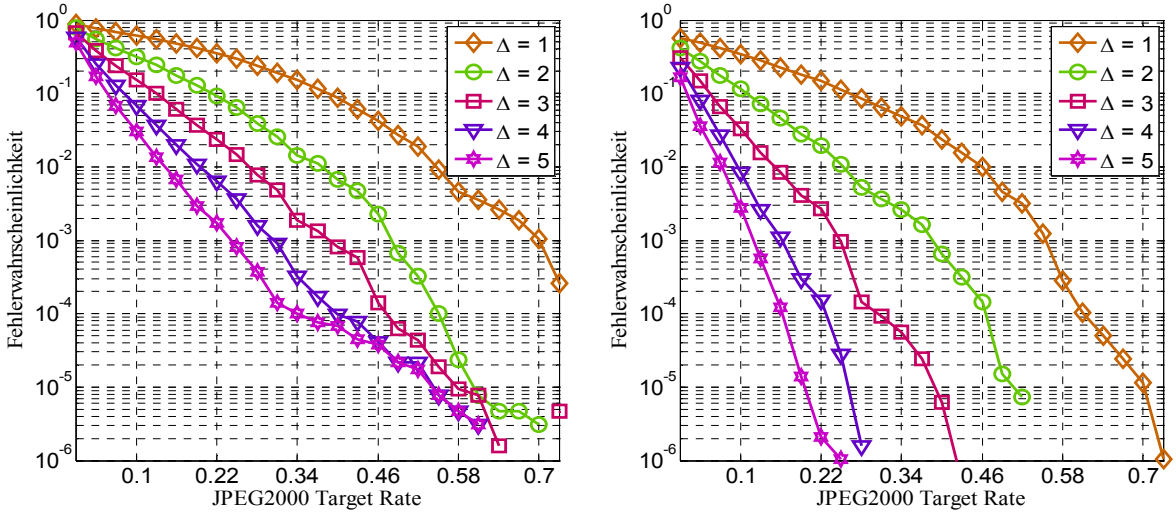


Abbildung B.45 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_3 (links) bzw. HL_3, LH_3, HH_3 (rechts) gegenüber JPEG2000-Kompression

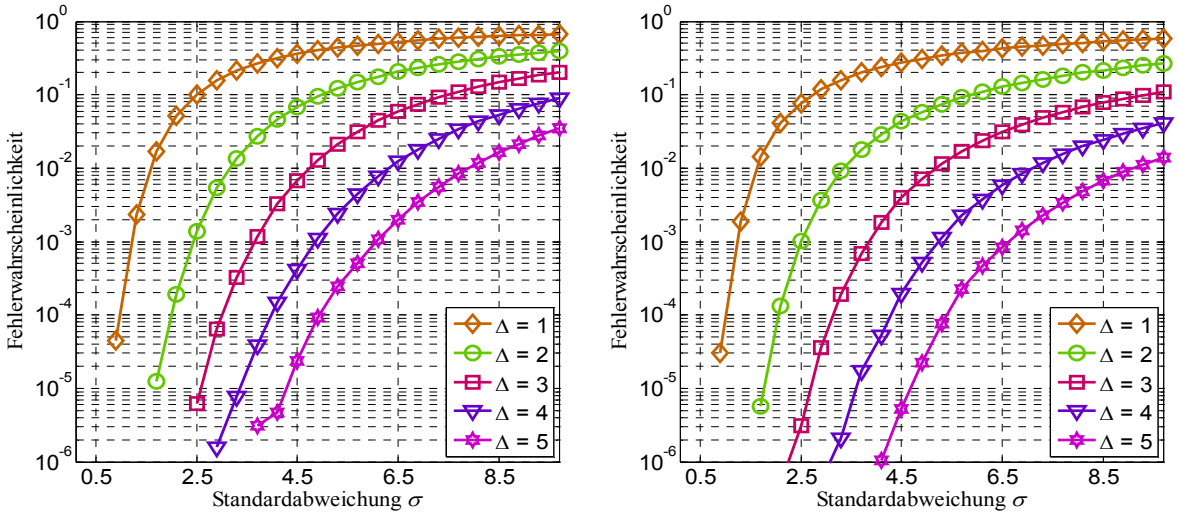


Abbildung B.46 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_3 (links) bzw. HL_3, LH_3, HH_3 (rechts) gegenüber Gaußschem Rauschen

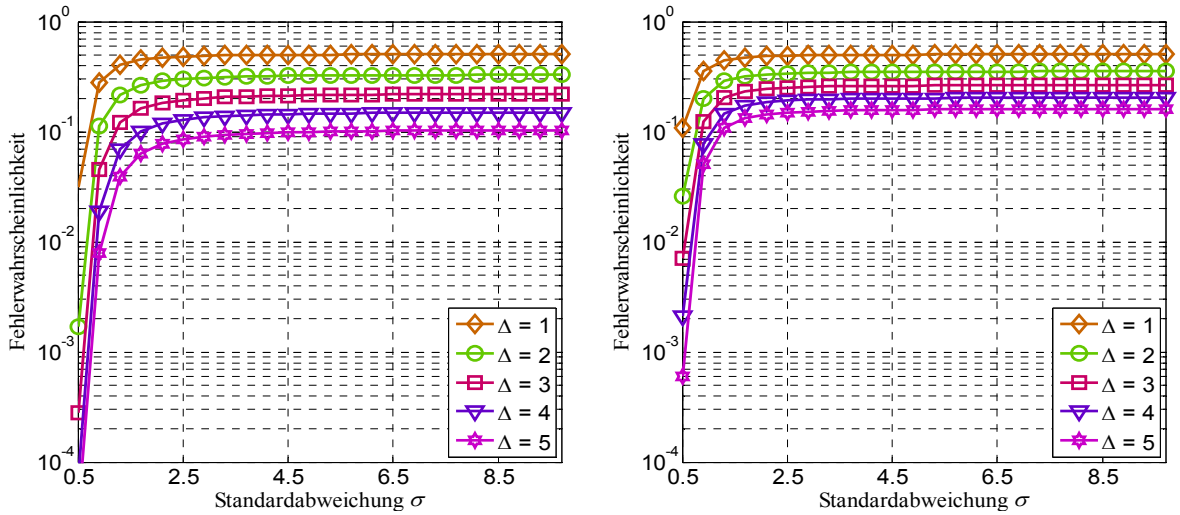


Abbildung B.47 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_3 (links) bzw. HL_3, LH_3, HH_3 (rechts) gegenüber Gaußscher Tiefpassfilterung

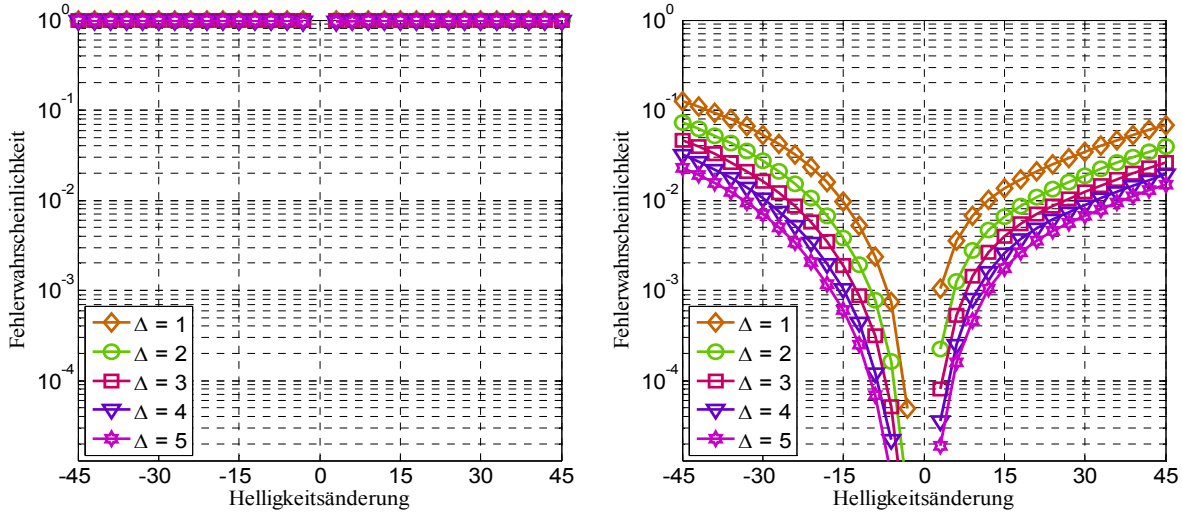


Abbildung B.48 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_3 (links) bzw. HL_3, LH_3, HH_3 (rechts) gegenüber Helligkeitsänderungen bezogen auf den Grauwertebereich $[0;255]$

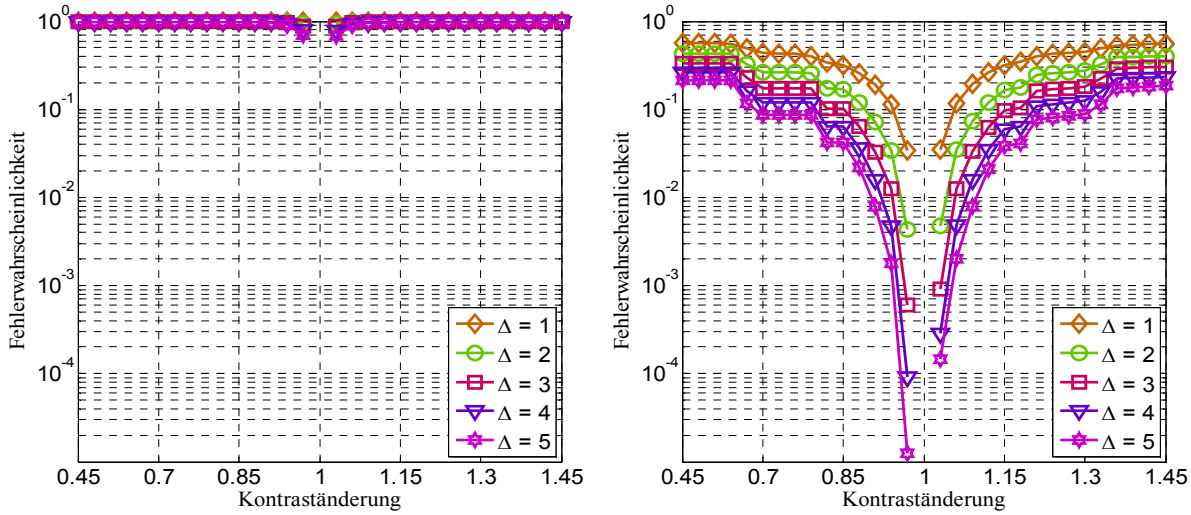


Abbildung B.49 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_3 (links) bzw. HL_3, LH_3, HH_3 (rechts) gegenüber Kontraständerungen bezogen auf den Grauwertebereich $[0;255]$

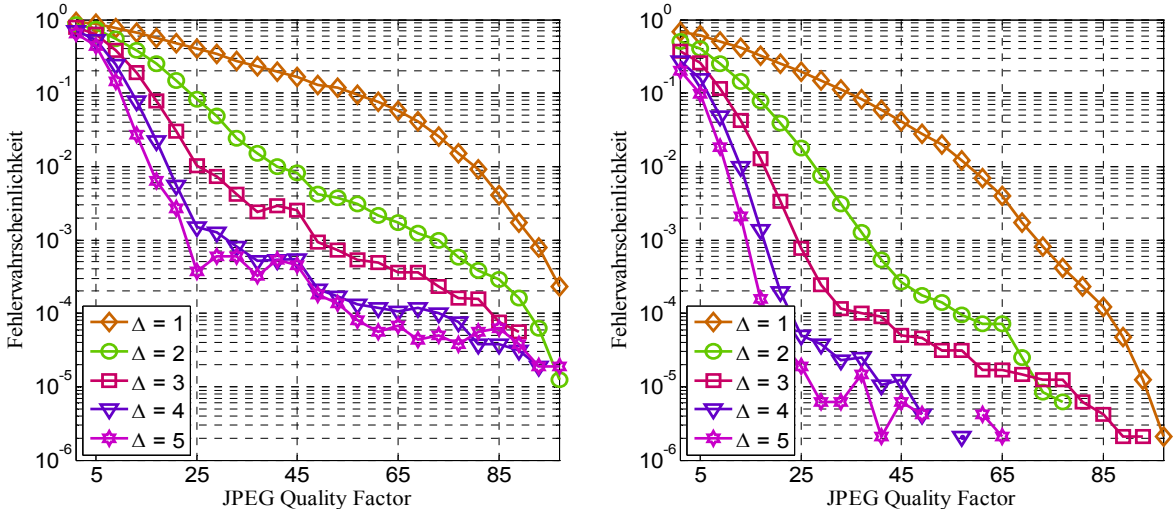


Abbildung B.50 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_4 (links) bzw. HL_4, LH_4, HH_4 (rechts) gegenüber JPEG-Kompression

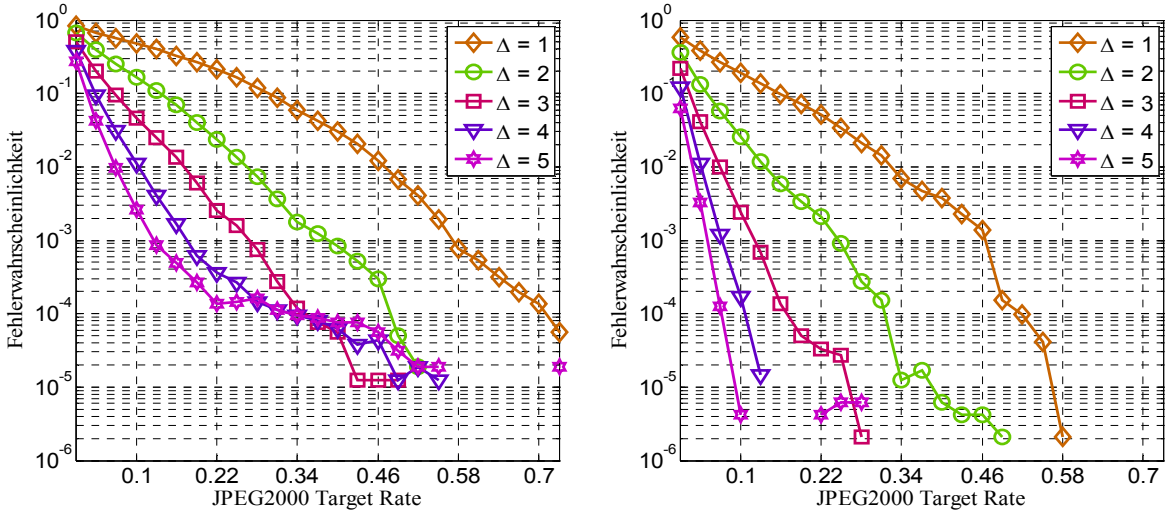


Abbildung B.51 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_4 (links) bzw. HL_4, LH_4, HH_4 (rechts) gegenüber JPEG2000-Kompression

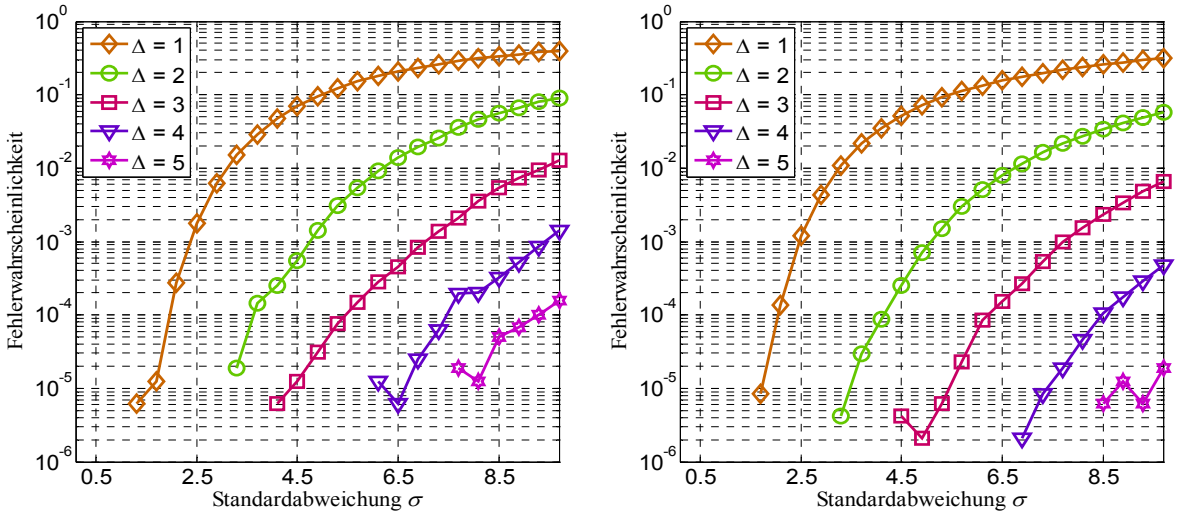


Abbildung B.52 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_4 (links) bzw. HL_4, LH_4, HH_4 (rechts) gegenüber Gaußschem Rauschen

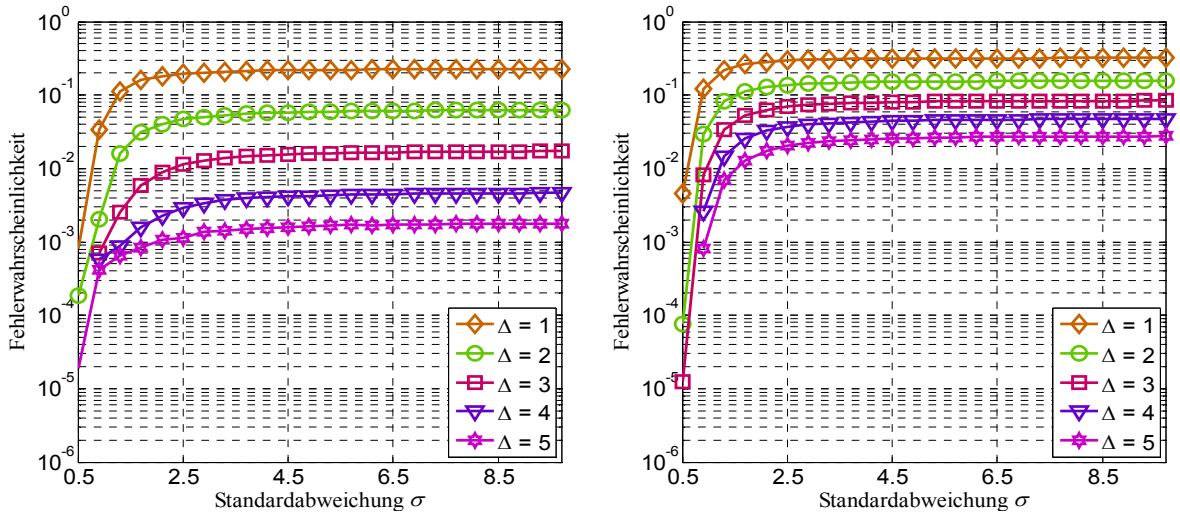


Abbildung B.53 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_4 (links) bzw. HL_4, LH_4, HH_4 (rechts) gegenüber Gaußscher Tiefpassfilterung

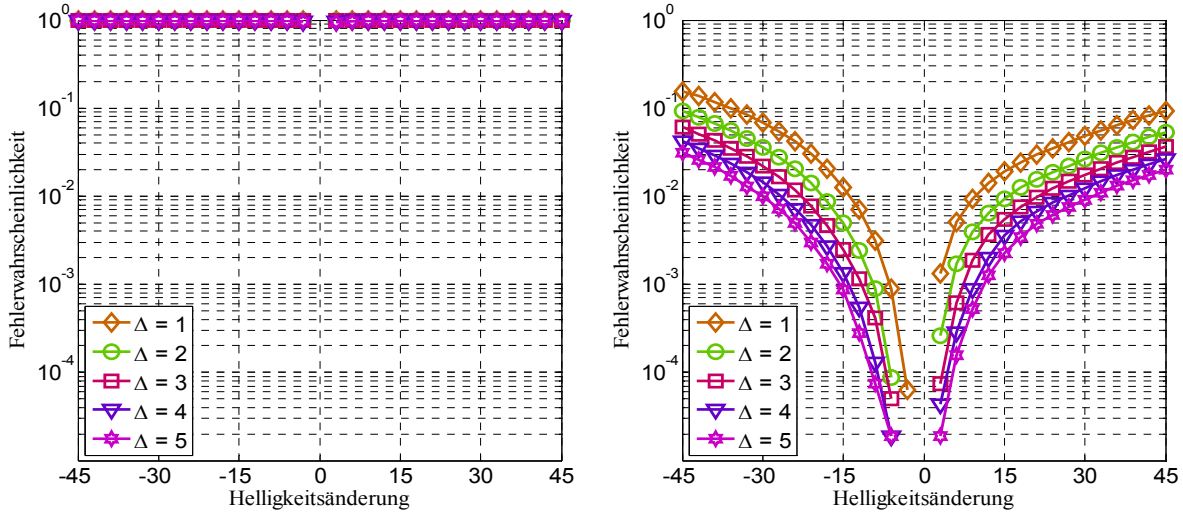


Abbildung B.54 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_4 (links) bzw. HL_4, LH_4, HH_4 (rechts) gegenüber Helligkeitsänderungen bezogen auf den Grauwertebereich $[0;255]$

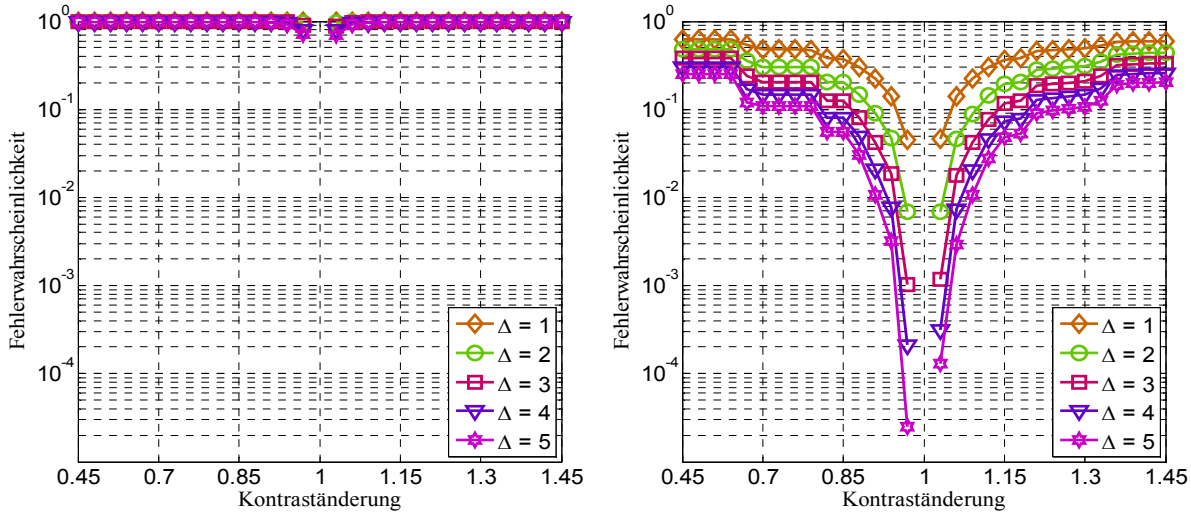


Abbildung B.55 Robustheit der quantisierten DWT-Koeffizienten des Subbandes LL_4 (links) bzw. HL_4, LH_4, HH_4 (rechts) gegenüber Kontraständerungen bezogen auf den Grauwertebereich $[0;255]$

Anhang C

Leistungsanalyse der nicht-adaptiven JPEG2000-basierten Authentifizierung

C.1 Bildverzerrungen durch die entwickelte Authentifizierung

In Abbildung C.1 sind die Bildverzerrungen dargestellt, die durch den Gesamtprozess der in Kapitel 3 beschrieben nicht-adaptiven JPEG2000-basierten Bildauthentifizierung entstehen.

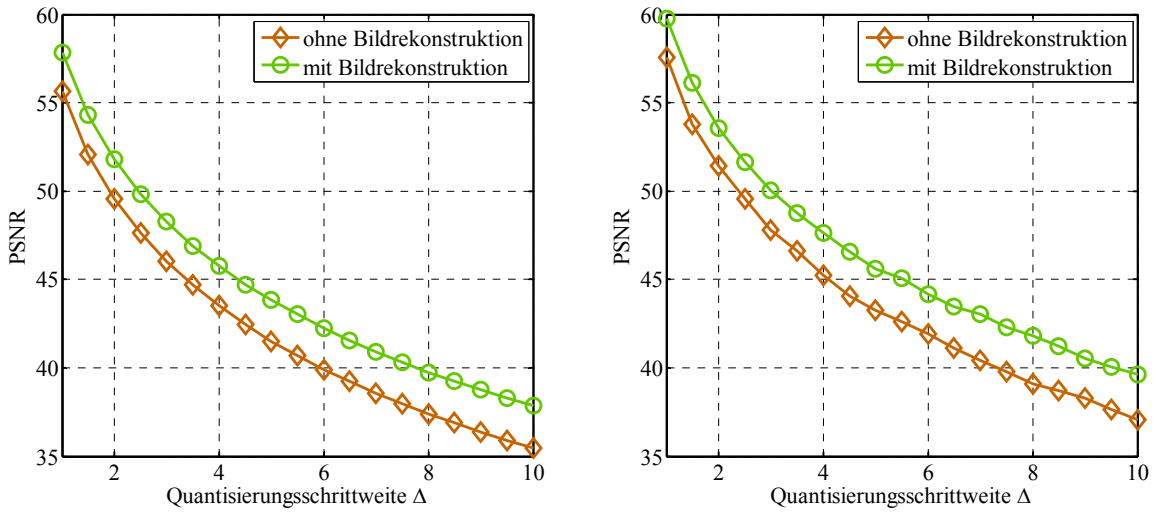


Abbildung C.1 Bildverzerrungen durch die Authentifizierung mit und ohne Bildrekonstruktion (Entfernung des Wasserzeichens). PSNR, gemittelt über 52 Testbilder (links), PSNR für das Testbild *Lena* "tb37" (rechts)

C.2 Robustheit der entwickelten Authentifizierung

In diesem Abschnitt wird die Robustheit der in Kapitel 3 entwickelten nicht-adaptiven JPEG2000-basierten Authentifizierung (mit Helligkeits-/Kontrastnormierung und Hash-Intervall-Fehlerkorrektur) gegenüber erlaubten, den Bildinhalt nicht verändernden Störungen untersucht. Dazu wurden 52 Testbilder bei unterschiedlichen Einbettungsstärken Δ mit einem Wasserzeichen versehen. Nach der Anwendung der Störungen wurden die Bilder auf Echtheit überprüft, wobei die Anzahl fehlgeschlagener Verifikationen in den nachfolgenden Abbildungen im Verhältnis zur Anzahl der gesamt getesteten Bilder dargestellt ist (*false positive ratio*).

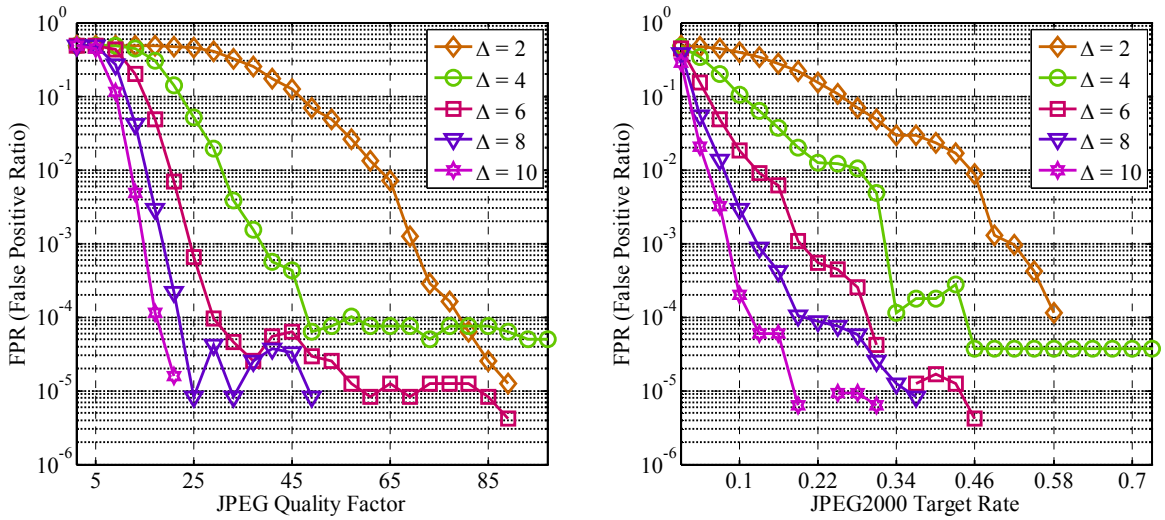


Abbildung C.2 Robustheit gegenüber JPEG-Kompression (links) und JPEG2000-Kompression (rechts)

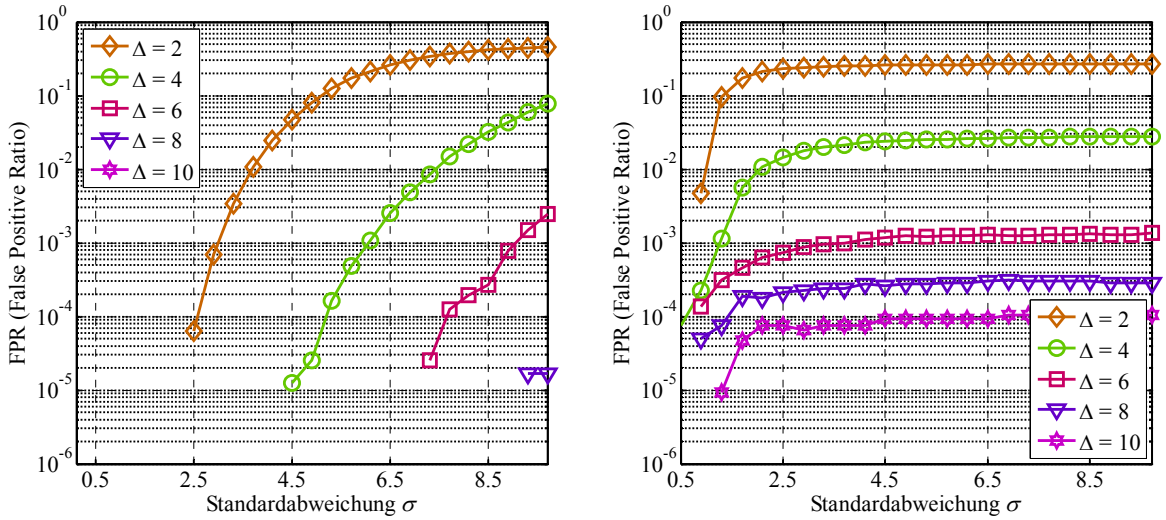


Abbildung C.3 Robustheit gegenüber Gaußschem Rauschen (links) und Gaußscher Tiefpassfilterung (rechts)

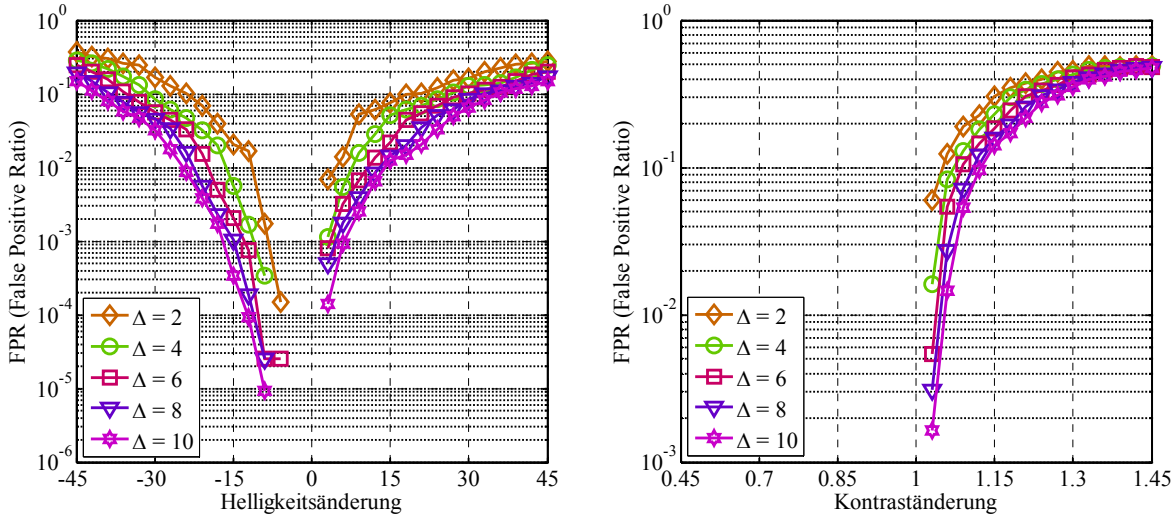


Abbildung C.4 Robustheit gegenüber Helligkeitsänderungen (links) und Kontraständerungen (rechts)

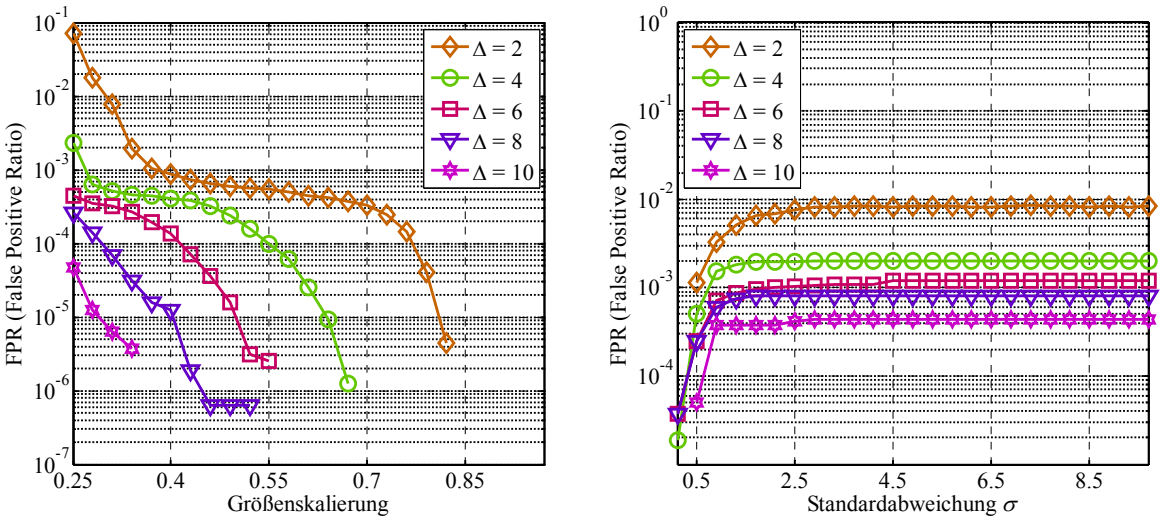


Abbildung C.5 Robustheit gegenüber Änderungen der Bildauflösung (links) und Bildschärfung (rechts)

C.3 Robustheit der Authentifizierung ohne Fehlerkorrektur des Hash-Intervalls

Durch das Einbetten des Authentifizierungswasserzeichens in den ebenfalls per Hash-Wert gesicherten DWT-Koeffizienten des LL_4-Subbandes sank die Robustheit der Hash-Intervall-Quantisierung für das ursprünglich entwickelte Authentifizierungsverfahren. Daraufhin wurde die Fehlerkorrektur der eingebetteten Wasserzeichenbits zu einer Fehlerkorrektur des jeweils berechneten Hash-Intervalls weiterentwickelt (siehe Abschnitt 3.2.3). In diesem Abschnitt sind zum Vergleich mit den Abbildungen des Abschnittes C.2 die Ergebnisse der Robustheitstests für die Authentifizierung ohne Hash-Intervall-Fehlerkorrektur dargestellt. Zur Simulation wurden dieselben Parameter verwendet wie im Abschnitt zuvor. Auf diese Weise ist eine Einschätzung der Leistungssteigerung durch die Hash-Intervall-Fehlerkorrektur möglich.

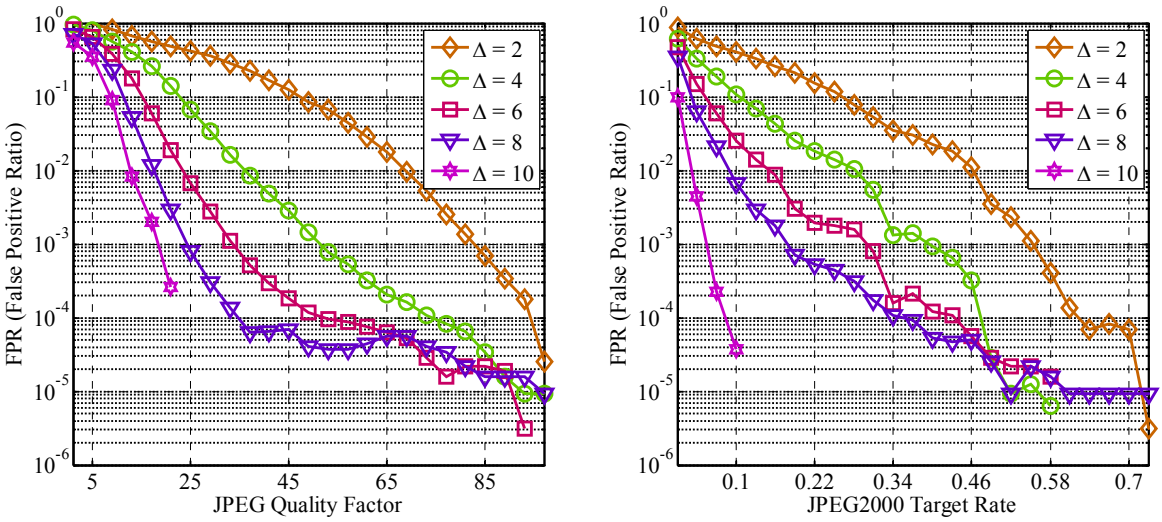


Abbildung C.6 Robustheit gegenüber JPEG-Kompression (links) und JPEG2000-Kompression (rechts)

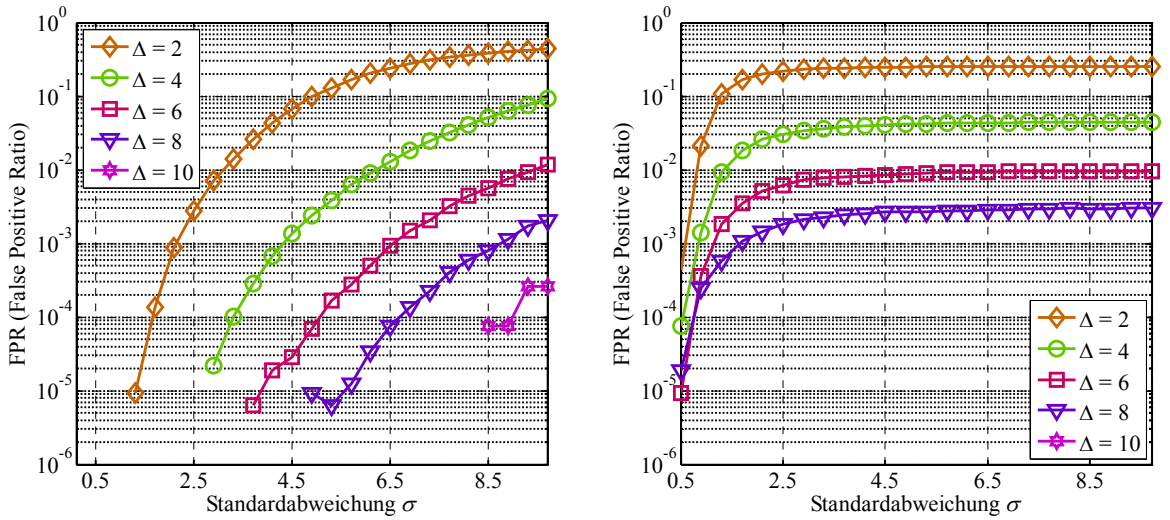


Abbildung C.7 Robustheit gegenüber Gaußschem Rauschen (links) und Gaußscher Tiefpassfilterung (rechts)

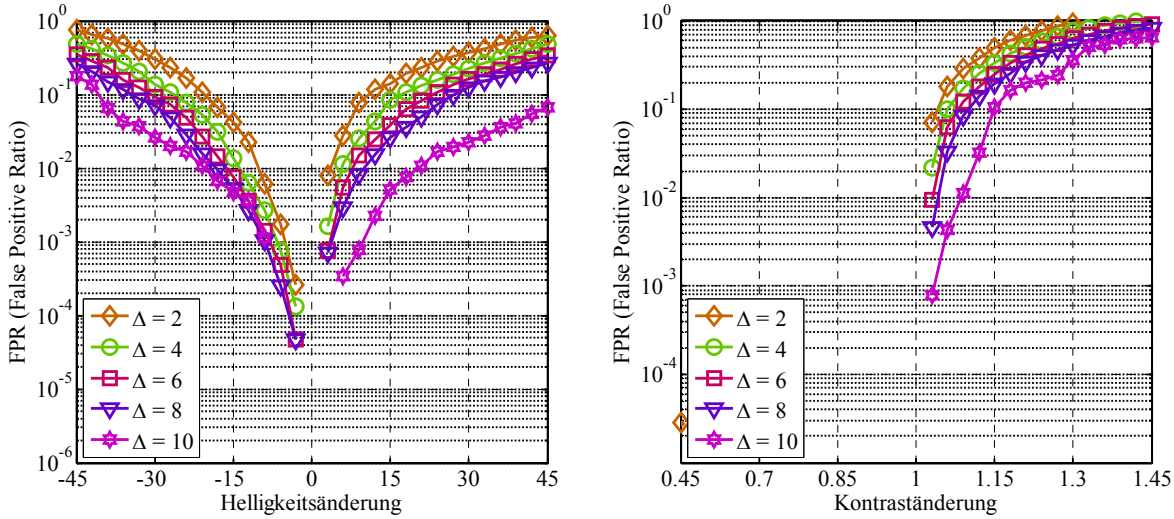


Abbildung C.8 Robustheit gegenüber Helligkeitsänderungen (links) und Kontraständerungen (rechts)

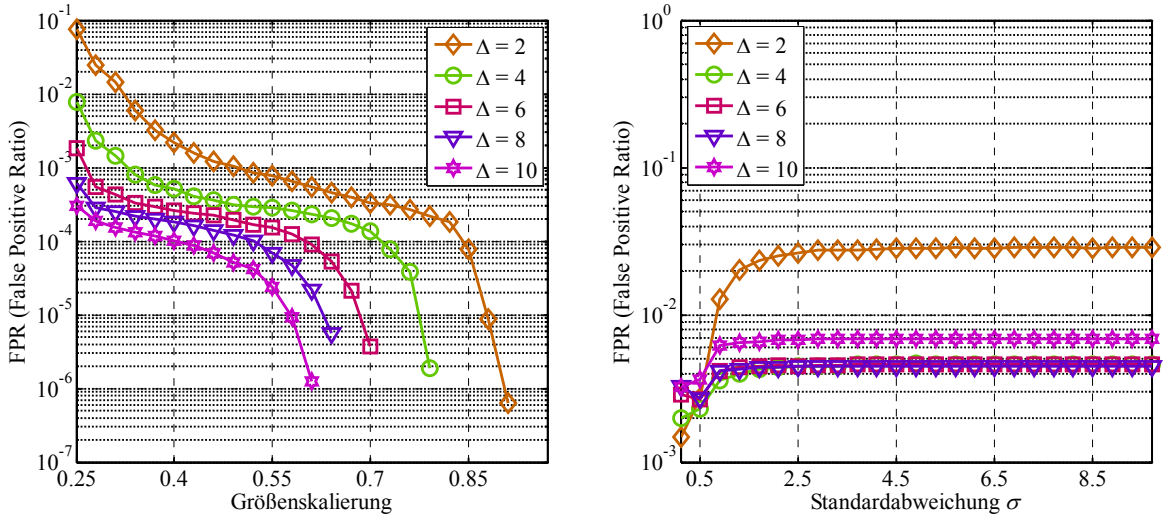


Abbildung C.9 Robustheit gegenüber Änderungen der Bildauflösung (links) und Bildschärfung (rechts)

Anhang D

Robustheit der Textursegmentierung

D.1 Textursegmentierung ohne Vorverzerrung

Die nachfolgenden Abbildungen zeigen die Robustheit[83] der in dieser Arbeit entwickelten Verfahren zur Unterteilung eines Bildes in Bereiche mit geringer und starker Textur ohne Vorverzerrung des Texturmerkmals. Neben dem in Abschnitt 4.2.1 beschriebenen gradientenbasierten (Grad-Seg) und DWT-basierten (DWT-Seg) Ansatz wurden auch die Verfahren von Huang *et al*. [HJJH01] (Huang-Seg) und Liu *et al*. [LHX01] (Liu-Seg) implementiert und unter gleichen Bedingungen getestet.

Für die 52 verwendeten Testbilder wurde die Segmentierungsmaske jeweils für 100 unterschiedliche[84] Schwellwerte τ bestimmt. Nach Anwendung der Störungen wurde die Maske erneut berechnet. Die Wahrscheinlichkeit des Auftretens von Maskenunterschieden (MER) ist dabei über die verwendeten Bilder und Schwellwerte gemittelt aufgetragen.

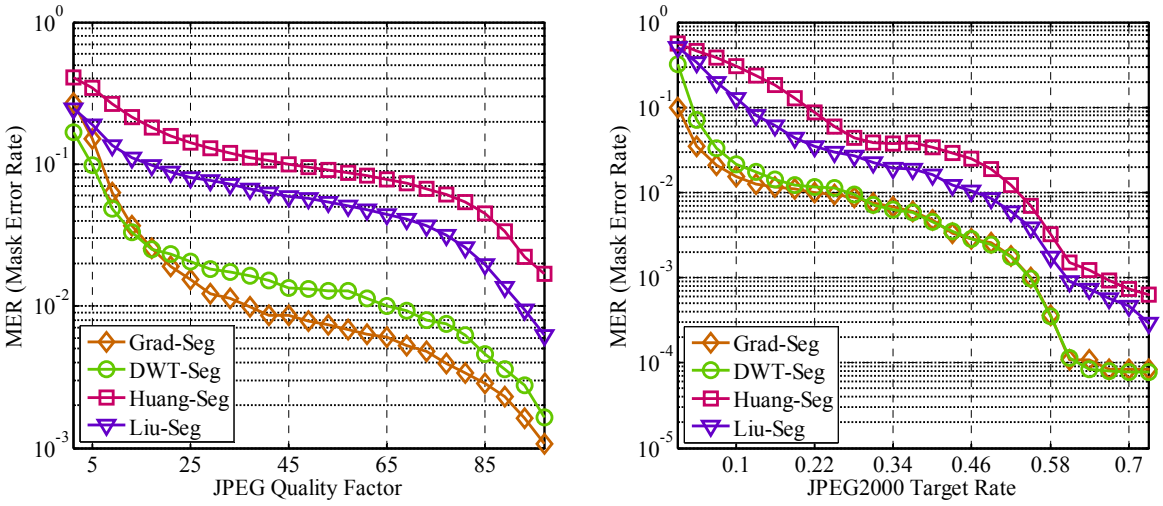


Abbildung D.1 Robustheit gegenüber JPEG-Kompression (links) und JPEG2000-Kompression (rechts)

[83] Fehlerwahrscheinlichkeit bei der Reproduktion der Bildunterteilung nach der Anwendung von Störungen.

[84] Die Wahl des Schwellwertes hat empirisch keinen merklichen Einfluss auf die Robustheit der Segmentierung. Durch die jeweils 100 leicht unterschiedlichen τ wird der Testsatz praktisch auf 52·100 Bilder vergrößert.

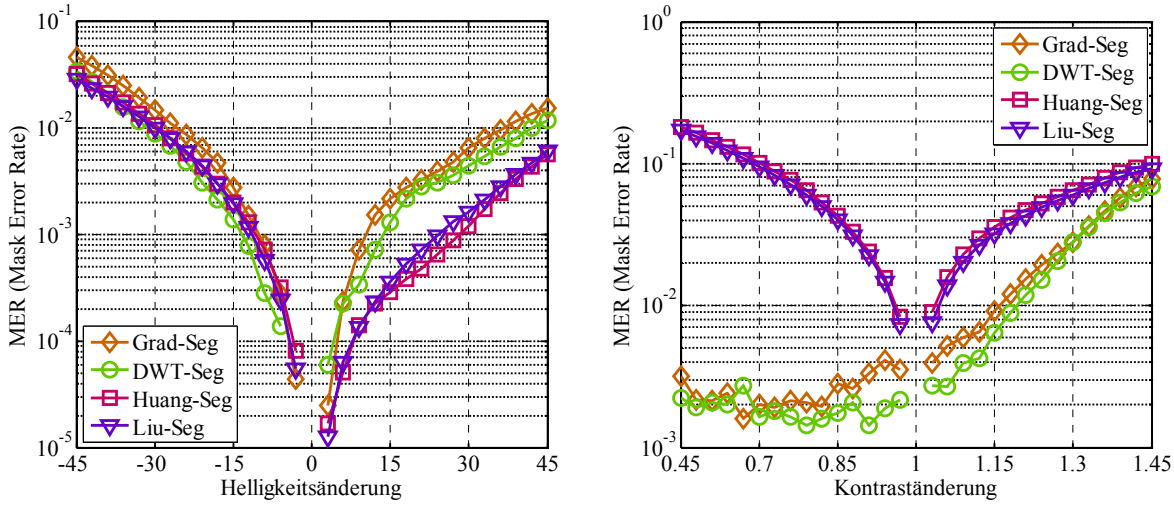


Abbildung D.2 Robustheit gegenüber Helligkeitsänderungen (links) und Kontraständerungen (rechts)

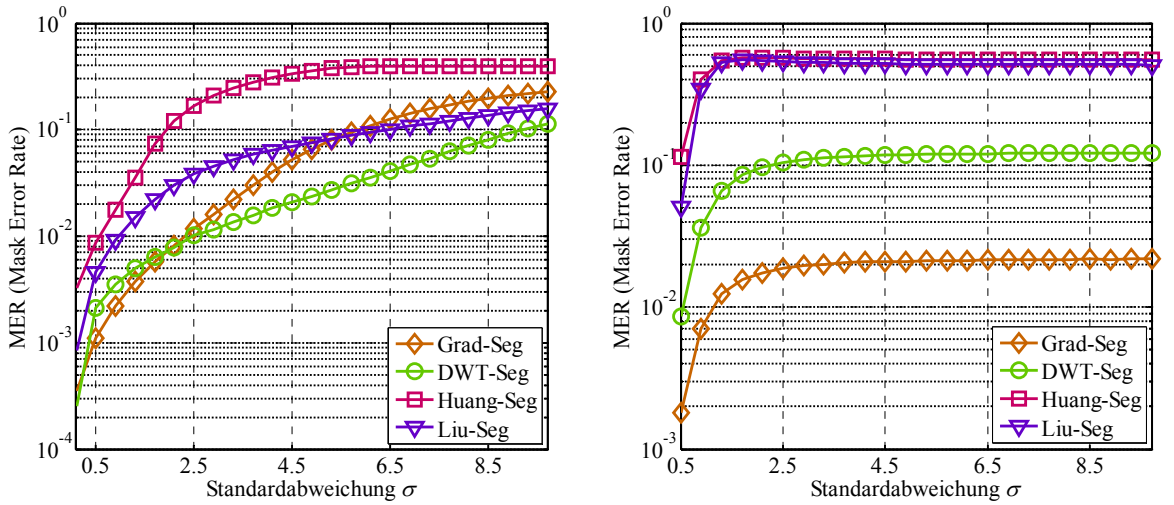


Abbildung D.3 Robustheit gegenüber Gaußschem Rauschen (links) und Gaußscher Tiefpassfilterung (rechts)

D.2 Textursegmentierung mit Vorverzerrung

Dieser Abschnitt demonstriert die Auswirkung einer Vorverzerrung des zur Segmentierung verwendeten Texturmerkmals für das in dieser Arbeit entwickelte DWT-basierte Verfahren zur texturabhängigen Bildunterteilung DWT-Seg (vgl. Abschnitt 4.2.1).

Auch hier wurde für die 52 Testbilder die Segmentierungsmaske jeweils für 100 unterschiedliche Schwellwerte τ bestimmt. Dabei wurde das verwendete Texturmerkmal, wie in Abschnitt 4.2.2 beschrieben, um den Wert t_τ vorverzerrt. Nach Anwendung der Störungen wurde die Maske erneut berechnet. Die Wahrscheinlichkeit des Auftretens von Maskenunterschieden (MER) ist dabei über die verwendeten Bilder und Schwellwerte gemittelt aufgetragen.

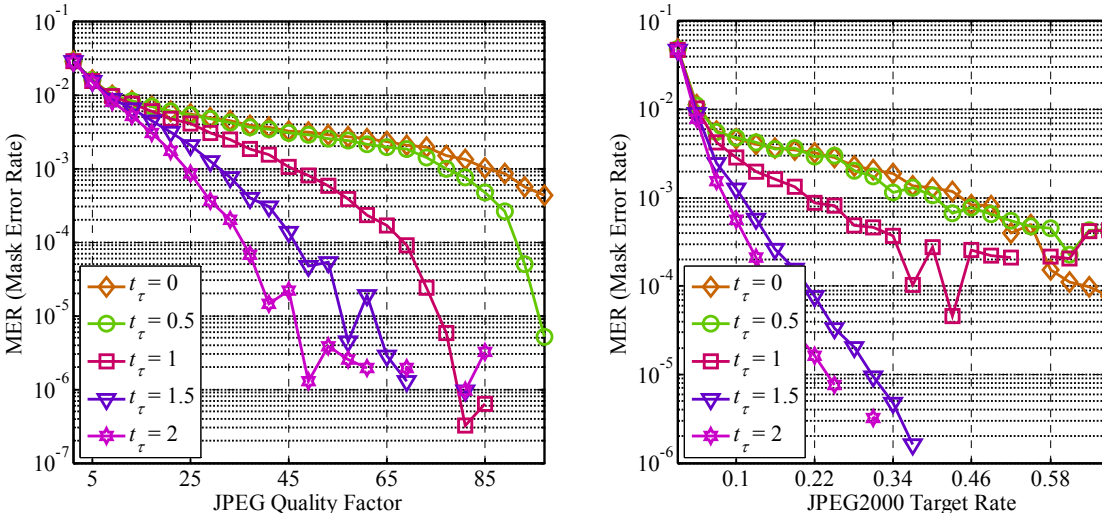


Abbildung D.4 Robustheit gegenüber JPEG-Kompression (links) und JPEG2000-Kompression (rechts)

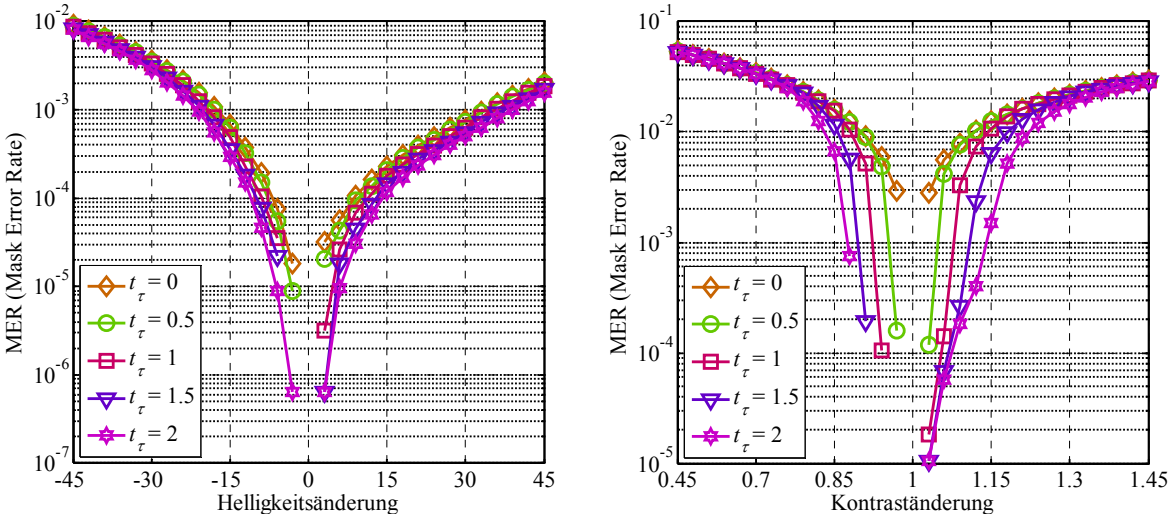


Abbildung D.5 Robustheit gegenüber Helligkeitsänderungen (links) und Kontraständerungen (rechts)

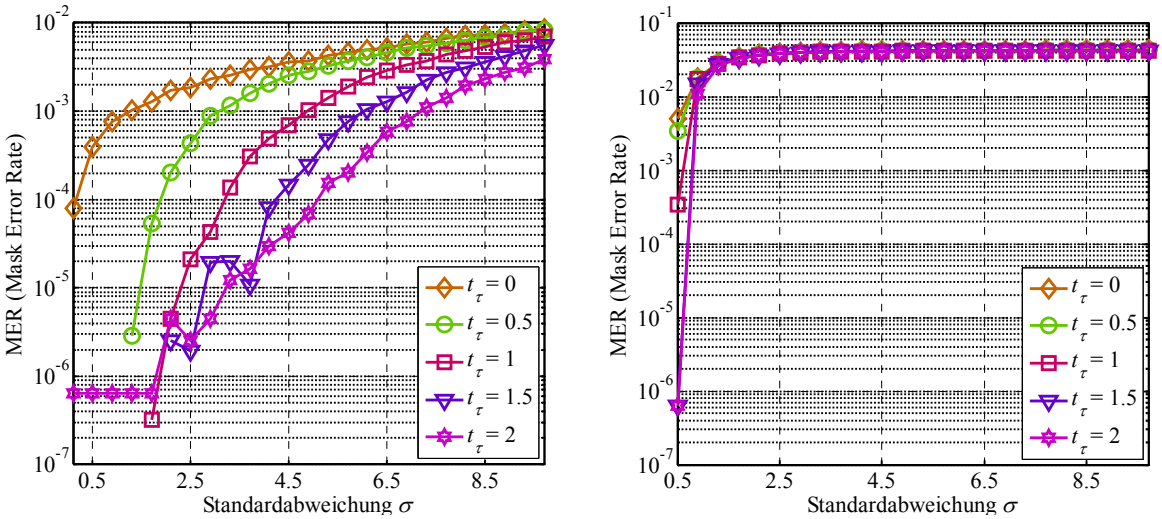


Abbildung D.6 Robustheit gegenüber Gaußschem Rauschen (links) und Gaußscher Tiefpassfilterung (rechts)

D.3 Auswirkungen der Vorverzerrungen auf die Bildqualität

Sollte bei dem in dieser Arbeit entwickelten DWT-basierten Verfahren zur Bildunterteilung die Vorverzerrung des Texturmerkmals um den Wert t_τ zum Einsatz kommen, sind Auswirkungen auf die Bildqualität zu erwarten. Anhand von Simulationen wurde untersucht, in welchem Ausmaß die Bildqualität durch die in Abschnitt 4.2.1 beschriebenen Vorverzerrungen beeinflusst wird. Folgende Abbildungen zeigen die Ergebnisse dieser Untersuchungen.

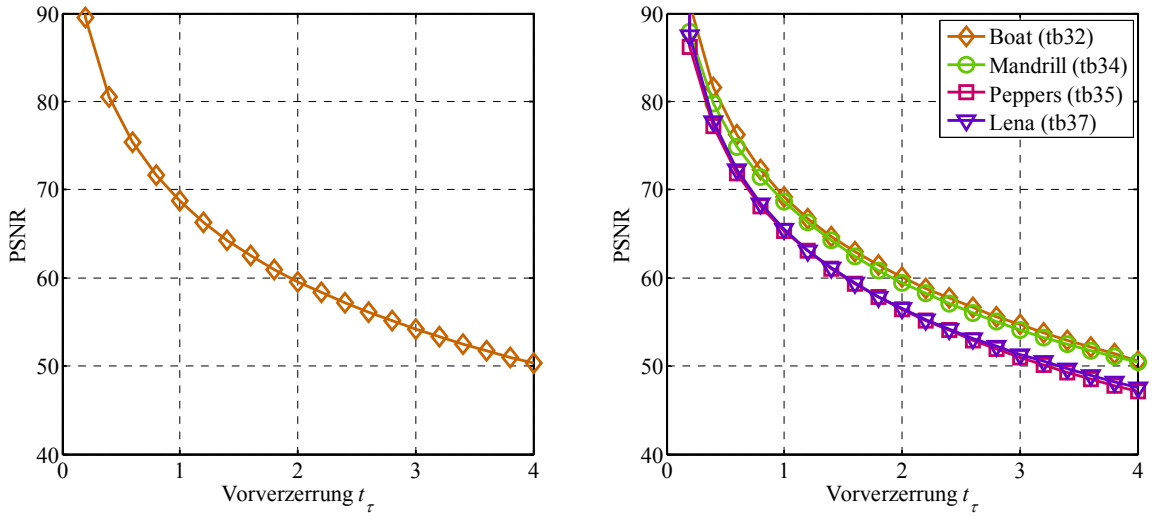


Abbildung D.7 Bildverzerrungen durch die Vorverzerrung des Texturmerkmals. PSNR gemittelt über 52 Testbilder (links), PSNR für die Testbilder *Boat* "tb32", *Mandrill* "tb34", *Peppers* "tb35" und *Lena* "tb37" (rechts)

Aus den Simulationsergebnissen sowie dem in der folgenden Abbildung D.8 dargestellten Beispiel eines vorverzerrten Bildes wird geschlussfolgert, dass die vorgeschlagene Vorverzerrung des Texturmerkmals bei $t_\tau = 2$ keine wahrnehmbaren Auswirkungen auf die Bildqualität zur Folge hat. In Anbetracht der in Abschnitt D.2 ermittelten deutlichen Verbesserung der Robustheit der Bildunterteilung ist ein Einsatz der Vorverzerrung für das in dieser Arbeit entwickelte Authentifizierungssystem denkbar.

Abbildung D.8 Beispiel: Veränderungen durch die Vorverzerrung des Texturmerkmals für die Bildsegmentierung - Original (links), Bild mit Vorverzerrung von $t_\tau = 2$ (mittig), kontrastverstärktes Differenzbild (rechts)

Anhang E

Leistungsanalyse der texturabhängigen adaptiven Wasserzeicheneinbettung

In diesem Kapitel des Anhangs geht es darum, die in Abschnitt 4.2 weiterentwickelte adaptive Einbettung mit der nicht-adaptiven Einbettung zu vergleichen. Ziel ist es, die Verbesserung der Wasserzeichenextraktion durch die in Abschnitt 4.2.3 vorgestellte Kombination von Bildsegmentierung und Fehlerkorrektur zu erfassen. Dazu wird die texturabhängige Einbettung in Abschnitt E.1 zunächst ohne die Quantisierung zur Berechung des bildabhängigen Hash-Wertes untersucht. Die Analyse des gesamten, erweiterten Authentifizierungssystems samt Hash-Wert-Generierung und -verifikation erfolgt anschließend in Abschnitt E.2.

E.1 Harte vs. weiche Bildsegmentierung

Die Abbildungen in diesem Abschnitt zeigen die Simulationsergebnisse hinsichtlich Robustheit der adaptiven Einbettung gegenüber Störungen bei der Verwendung unterschiedlicher Quantisierungsstärken. Die Unterteilung der dabei verwendeten 52 Testbilder erfolgte durch die DWT-basierte Textursegmentierung aus Abschnitt 4.2.1 (ohne Vorverzerrung des Texturmerkmals). Zur Einbettung eines zufällig generierten 1024 Bit langen Wasserzeichens[85] kam das in Abschnitt 4.2 beschriebene adaptive Wasserzeichenverfahren mit den Quantisierungsschrittweiten Δ_1 und Δ_2 zum Einsatz (wobei $\Delta_2/\Delta_1 = 3$).

Die Kurven repräsentieren das Bitfehlerverhältnis nach der Extraktion. Kurve EMHD entspricht dabei der Extraktion unter Verwendung der Einbettungsmaske (Ausschluss von Maskenänderungen). EMHD stellt damit die Referenzkurve dar. HMHD und HMSD stehen für die Segmentierung durch die harte, auf der Extraktionsseite berechnete Maske. SMHD und SMSD wiederum entsprechen der Segmentierung durch die weiche, auf der Extraktionsseite berechnete Maske. Dabei bedeuten die beiden letzten Buchstaben SD und HD jeweils weiche bzw. harte (soft input zu ±1 quantisierte) FEC-Decodierung.

[85] ersetzt in dieser Simulation die 1024 Bit lange, mittels Hash-Funktion generierte, bildabhängige Signatur

E.1.1 Adaptive Einbettung mit $\Delta_2/\Delta_1 = 3$ ohne Textur-Vorverzerrung

Parameter der Simulation: $\Delta_1 = 2$, $\Delta_2 = 6$, $\tau = 1.5$, $\alpha = 5$

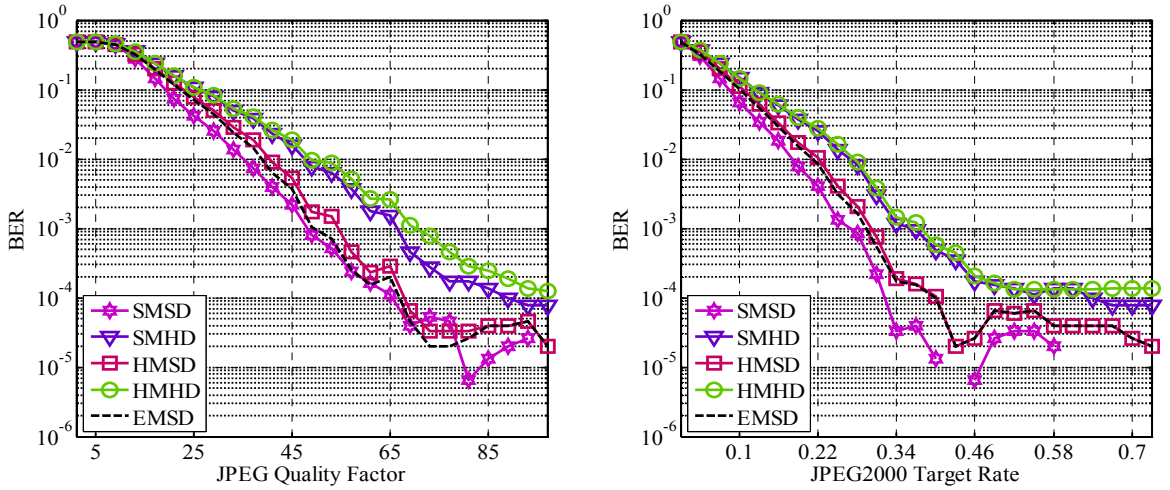


Abbildung E.1 Robustheit gegenüber JPEG-Kompression (links) und JPEG2000-Kompression (rechts)

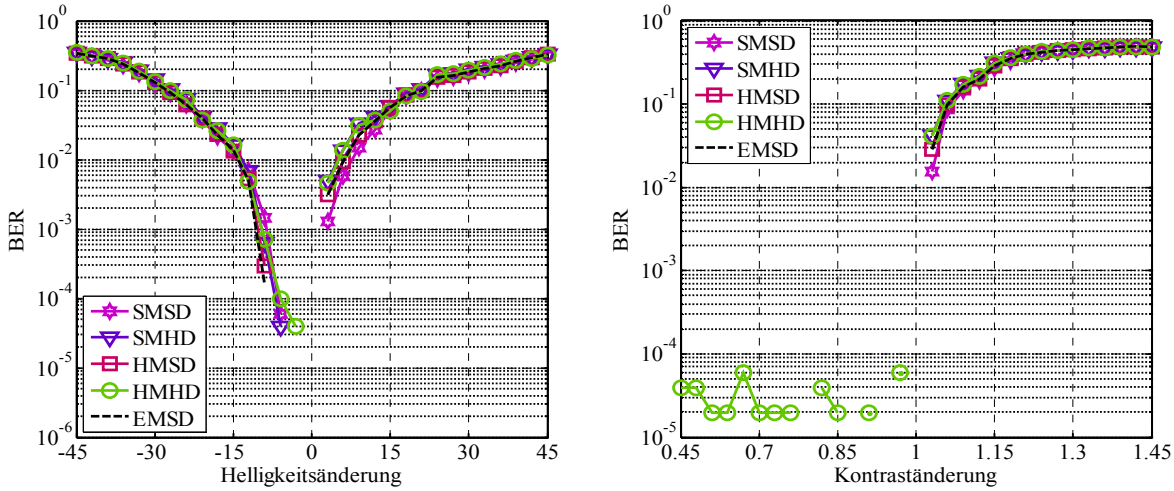


Abbildung E.2 Robustheit gegenüber Helligkeitsänderungen (links) und Kontraständerungen (rechts)

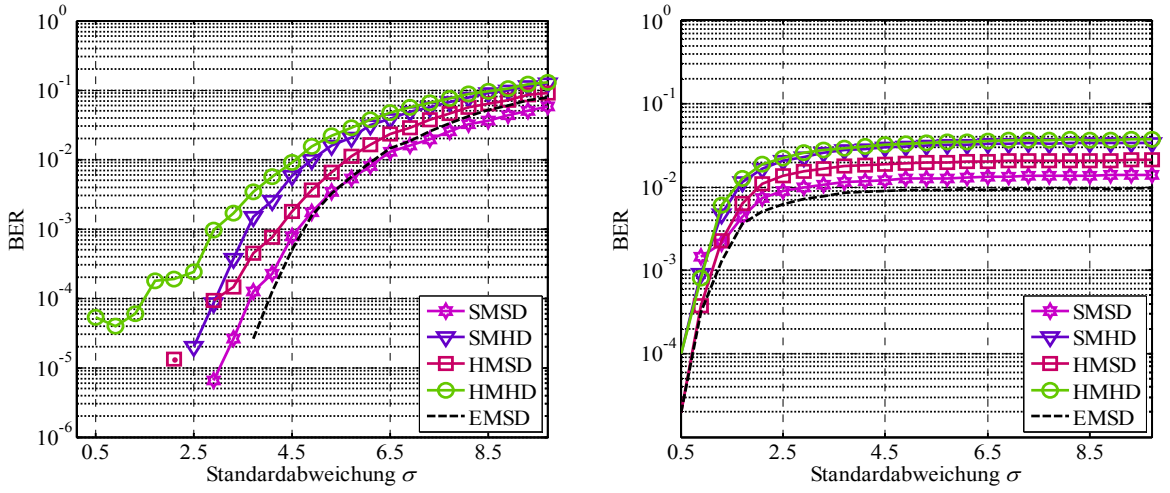


Abbildung E.3 Robustheit gegenüber Gaußschem Rauschen (links) und Gaußscher Tiefpassfilterung (rechts)

Parameter der Simulation: $\Delta_1 = 3$, $\Delta_2 = 9$, $\tau = 1.5$, $\alpha = 5$

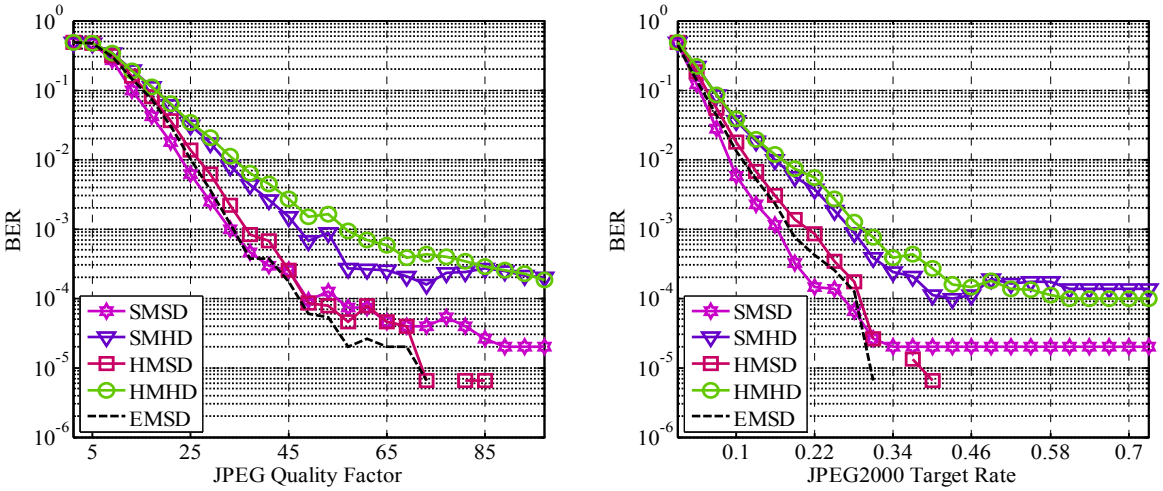


Abbildung E.4 Robustheit gegenüber JPEG-Kompression (links) und JPEG2000-Kompression (rechts)

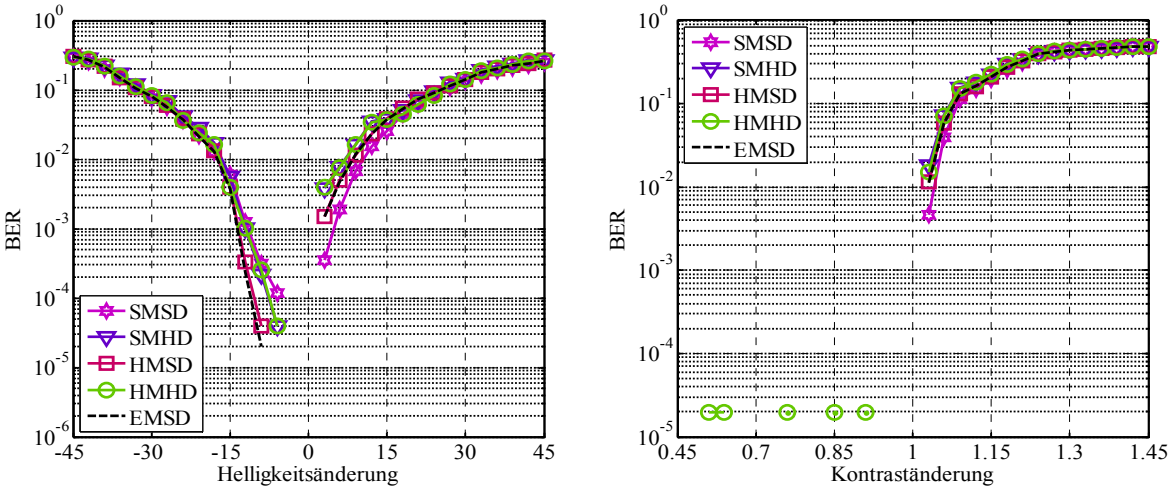


Abbildung E.5 Robustheit gegenüber Helligkeitsänderungen (links) und Kontraständerungen (rechts)

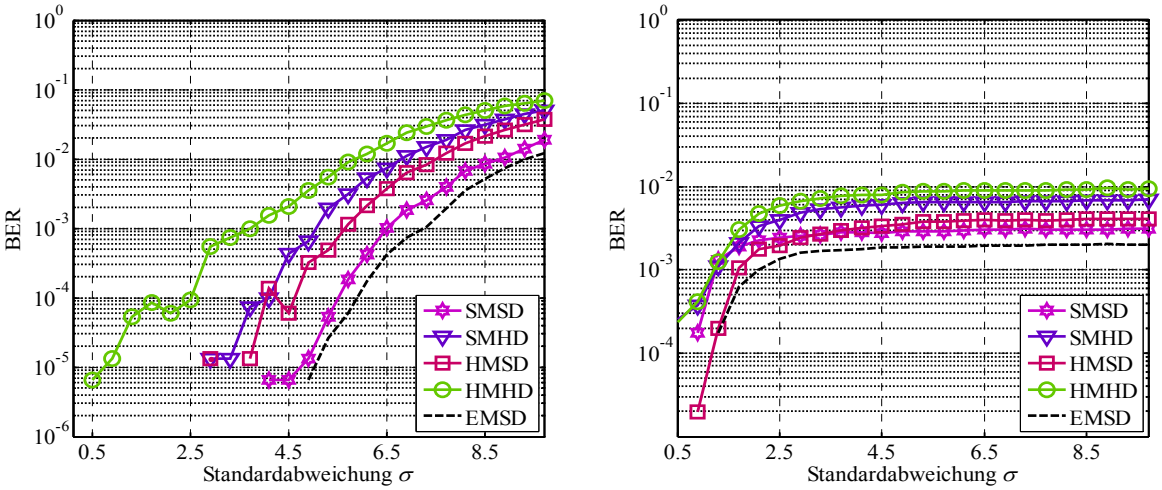


Abbildung E.6 Robustheit gegenüber Gaußschem Rauschen (links) und Gaußscher Tiefpassfilterung (rechts)

Parameter der Simulation: $\Delta_1 = 4$, $\Delta_2 = 12$, $\tau = 1.5$, $\alpha = 5$

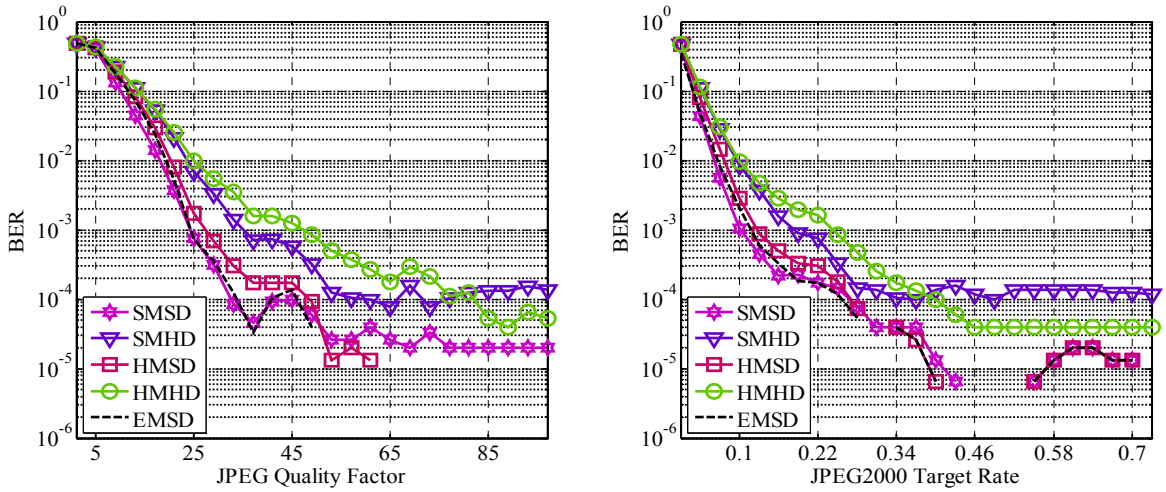


Abbildung E.7 Robustheit gegenüber JPEG-Kompression (links) und JPEG2000-Kompression (rechts)

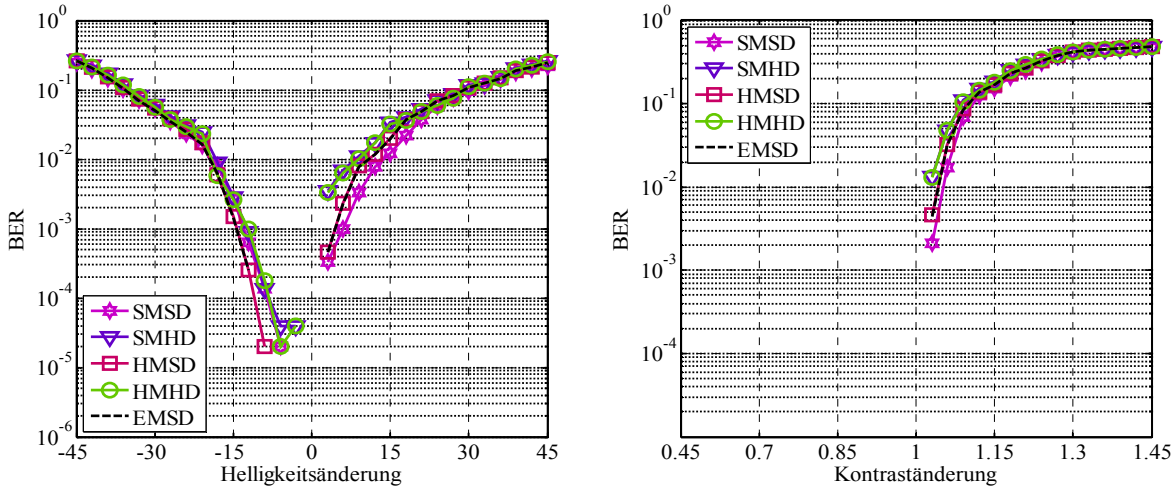


Abbildung E.8 Robustheit gegenüber Helligkeitsänderungen (links) und Kontraständerungen (rechts)

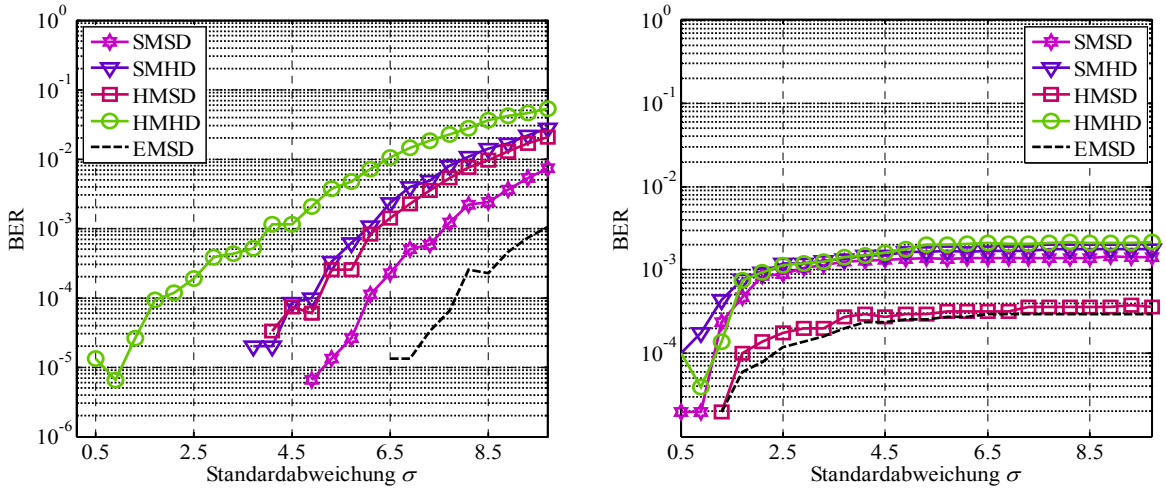


Abbildung E.9 Robustheit gegenüber Gaußschem Rauschen (links) und Gaußscher Tiefpassfilterung (rechts)

E.1.2 Adaptive Einbettung bei unterschiedlichen Verhältnissen der Quantisierungsschrittweiten ohne Textur-Vorverzerrung

In den folgenden Abbildungen sind die Bitfehlerverhältnisse der extrahierten Wasserzeichensequenzen bei der Verwendung unterschiedlicher Verhältnisse Δ_2 / Δ_1 dargestellt. Die kleinere Quantisierungsschrittweite Δ_1 bleibt dabei jeweils konstant.

Die nicht-adaptive Einbettung bei gleicher subjektiver Wahrnehmbarkeit der verursachten Bildverzerrungen (vgl. Abschnitt 4.2.4) entspricht hier dem Verhältnis $\Delta_2 / \Delta_1 = 1$. Dadurch wird ein direkter Vergleich zwischen nicht-adaptiver und adaptiver Einbettung ermöglicht.

Parameter der Simulation: $\tau = 1.5$, $\alpha = 5$

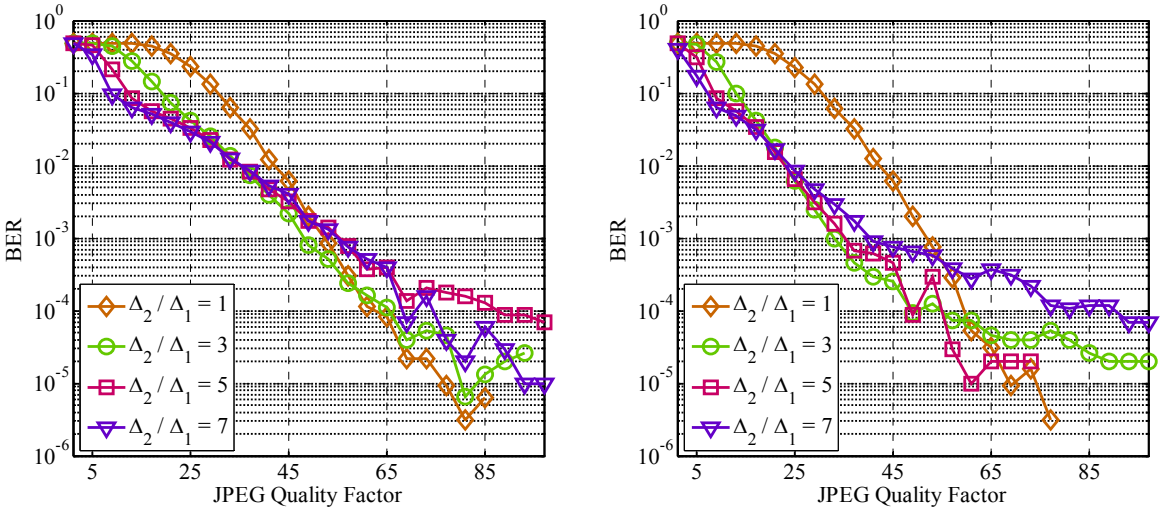


Abbildung E.10 Robustheit gegenüber JPEG-Kompression, wobei $\Delta_1 = 2$ (links) bzw. $\Delta_1 = 3$ (rechts)

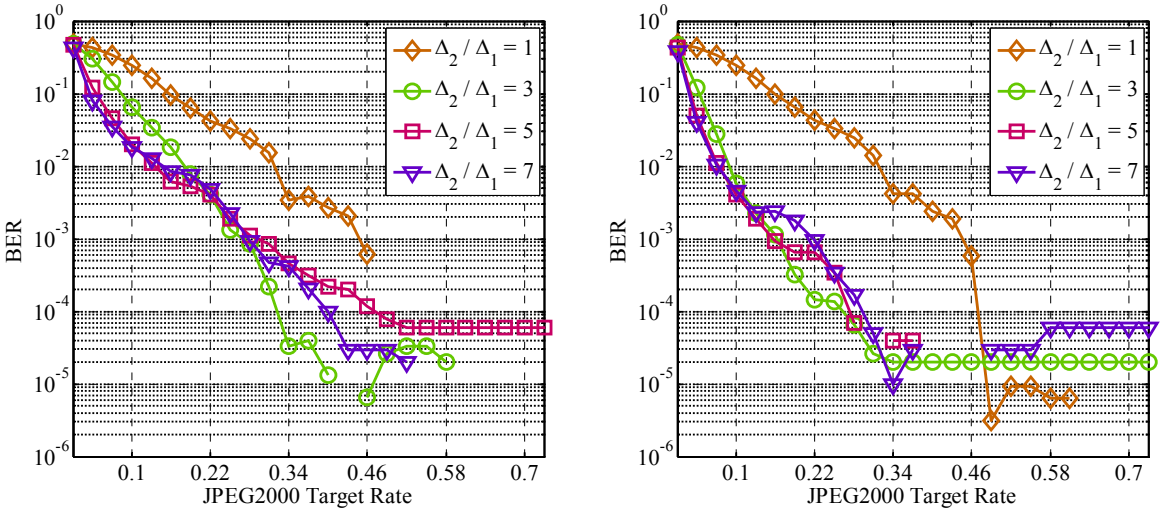


Abbildung E.11 Robustheit gegenüber JPEG2000-Kompression, wobei $\Delta_1 = 2$ (links) bzw. $\Delta_1 = 3$ (rechts)

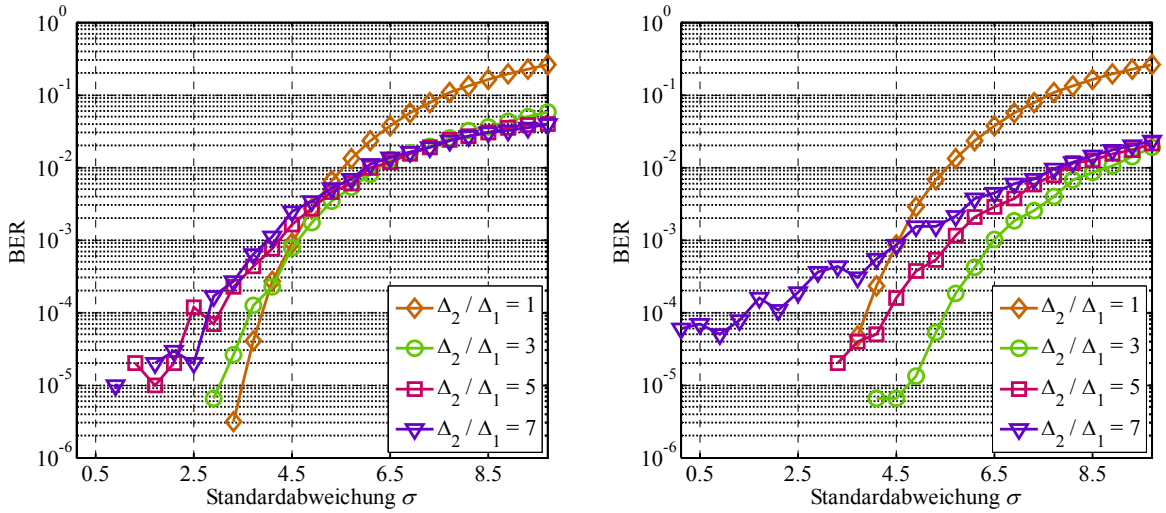


Abbildung E.12 Robustheit gegenüber Gaußschem Rauschen, wobei $\Delta_1 = 2$ (links) bzw. $\Delta_1 = 3$ (rechts)

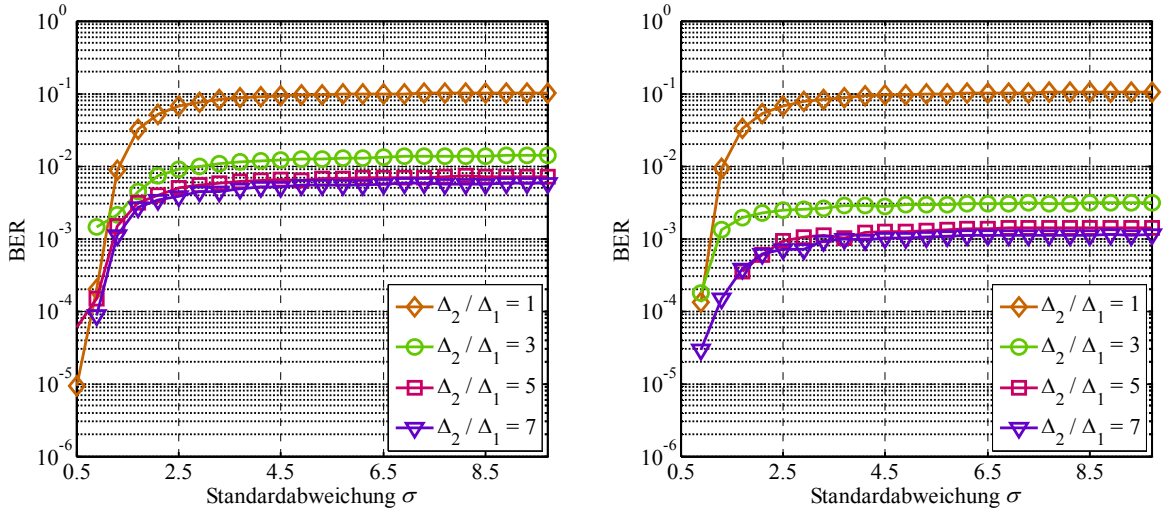


Abbildung E.13 Robustheit gegenüber Gaußscher Tiefpassfilterung, wobei $\Delta_1 = 2$ (links) bzw. $\Delta_1 = 3$ (rechts)

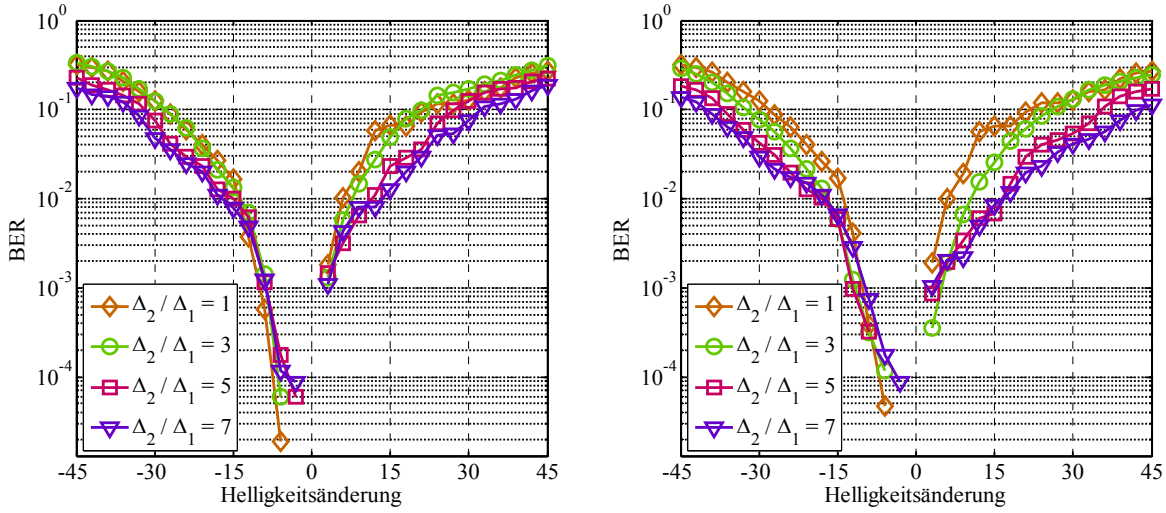


Abbildung E.14 Robustheit gegenüber Helligkeitsänderungen, wobei $\Delta_1 = 2$ (links) bzw. $\Delta_1 = 3$ (rechts)

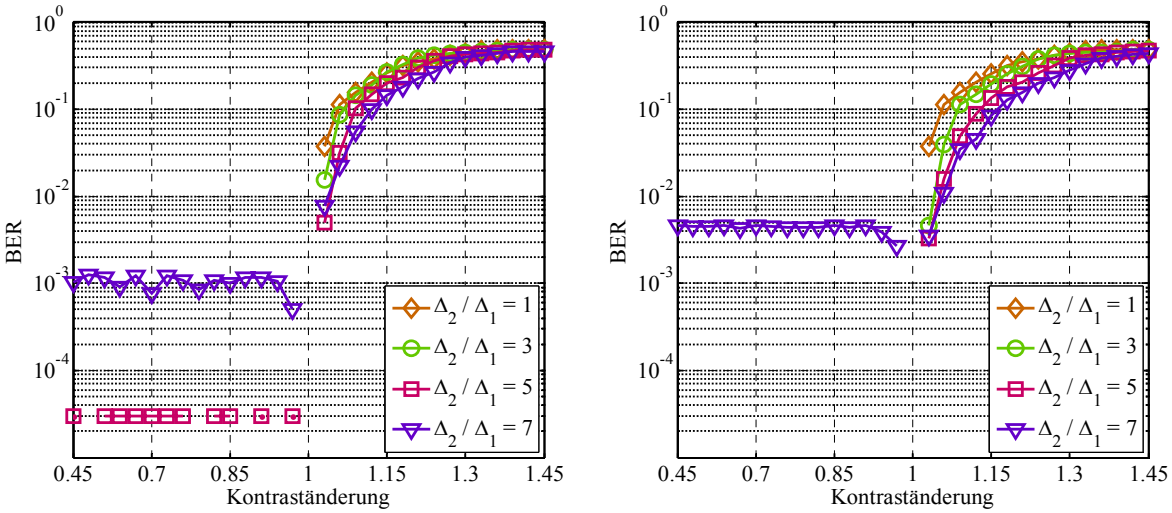


Abbildung E.15 Robustheit gegenüber Kontraständerungen, wobei $\Delta_1 = 2$ (links) bzw. $\Delta_1 = 3$ (rechts)

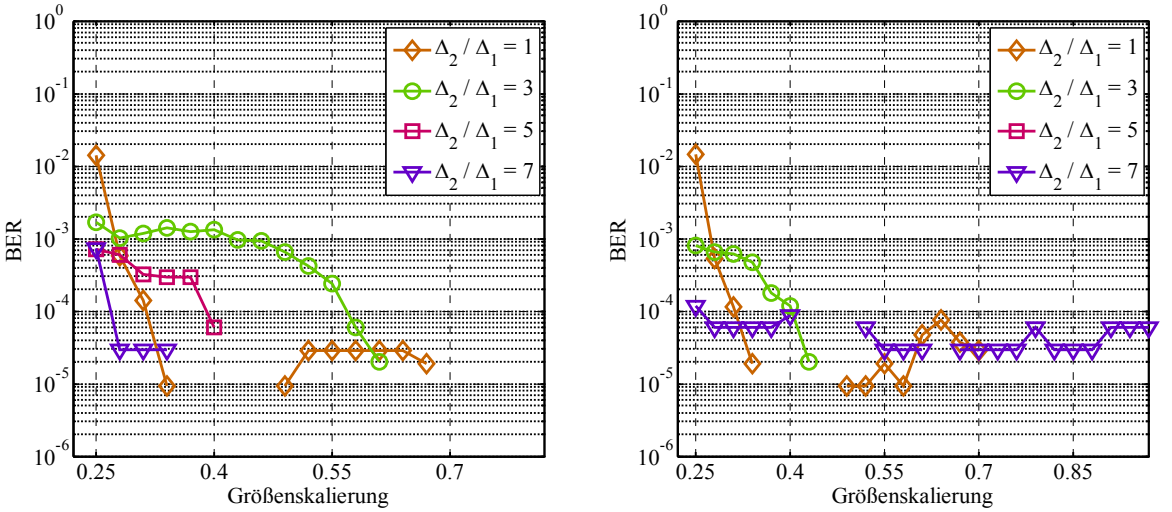


Abbildung E.16 Robustheit gegenüber Skalierungen der Bildgröße, wobei $\Delta_1 = 2$ (links) bzw. $\Delta_1 = 3$ (rechts)

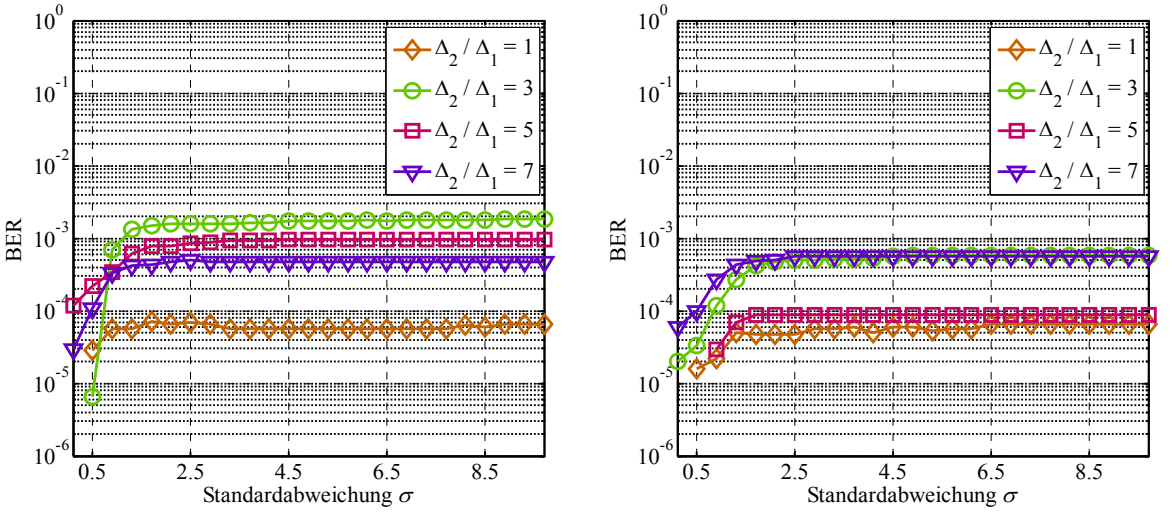


Abbildung E.17 Robustheit gegenüber Bildschärfung mittels Unsharp-Masking (Filtergröße 3x3), wobei $\Delta_1 = 2$ (links) bzw. $\Delta_1 = 3$ (rechts)

E.2 Robustheit der erweiterten adaptiven Authentifizierung

Die nachfolgenden Abbildungen zeigen die Ergebnisse der Robustheitssimulation des in Abschnitt 4.2 beschrieben, erweiterten, adaptiven Authentifizierungsverfahrens. Zum Vergleich sind die Simulationsergebnisse der bereits in Abschnitt C.2 aufgeführten nicht-adaptiven Authentifizierung (Kapitel 3) dargestellt. So lässt sich die durch die Weiterentwicklung erzielte Verbesserung direkt erkennen.

Zur Wasserzeicheneinbettung wurde die Quantisierungsschrittweite Δ_1 auf den Wert 2 bzw. 3 fixiert. Beim nicht-adaptiven Verfahren ist das Verhältnis $\Delta_2 / \Delta_1 = 1$. Für das adaptive Verfahren wurde das Verhältnis $\Delta_2 / \Delta_1 = 3$ gewählt. Die Bildunterteilung erfolgte durch das in Abschnitt 4.2.1 beschriebene DWT-basierte Textursegmentierungsverfahren mit dem Schwellwert $\tau = 1{,}5$. Für die Wichtung der adaptiven Wasserzeichenextraktion wurde $\alpha = 5$ gewählt.

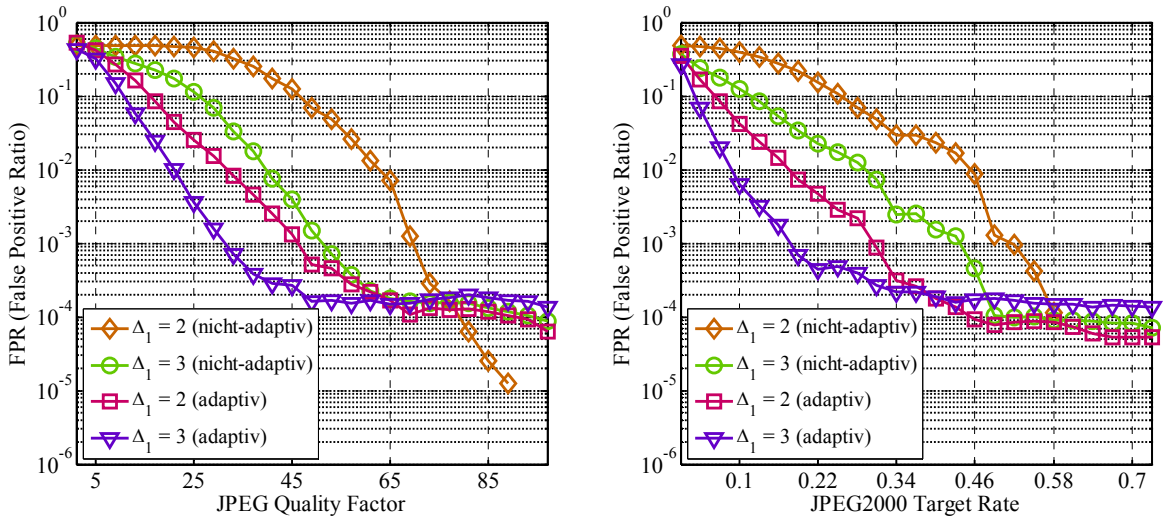


Abbildung E.18 Robustheit gegenüber JPEG-Kompression (links) und JPEG2000-Kompression (rechts)

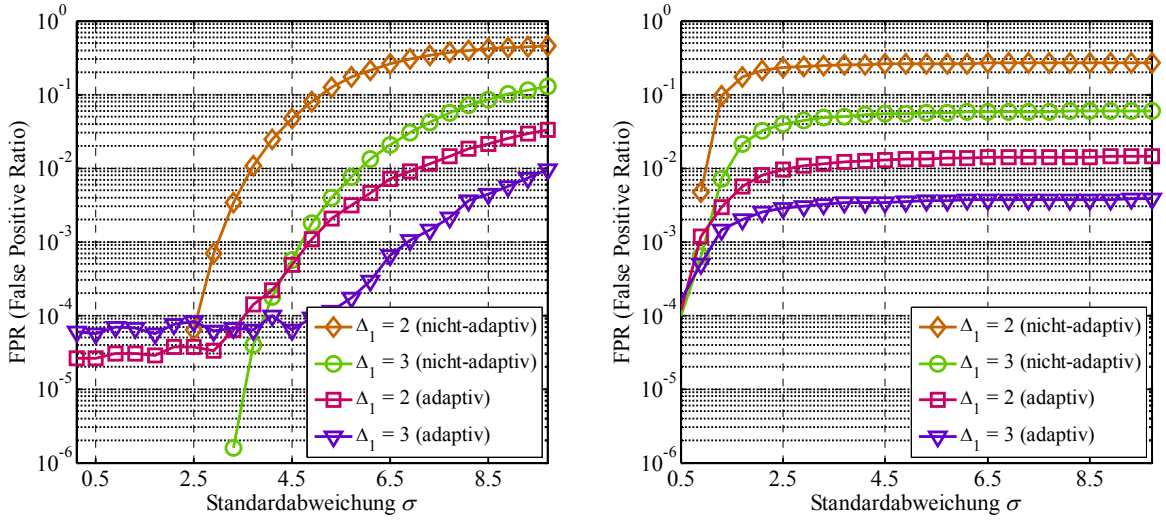


Abbildung E.19 Robustheit gegenüber Gaußschem Rauschen (links) und Gaußscher Tiefpassfilterung (rechts)

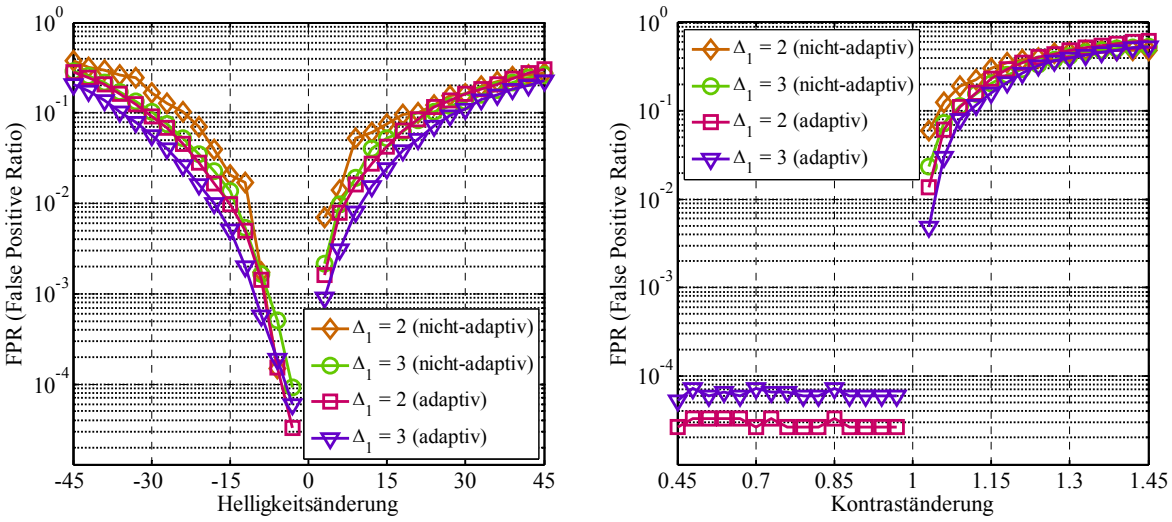


Abbildung E.20 Robustheit gegenüber Helligkeitsänderungen (links) und Kontraständerungen (rechts)

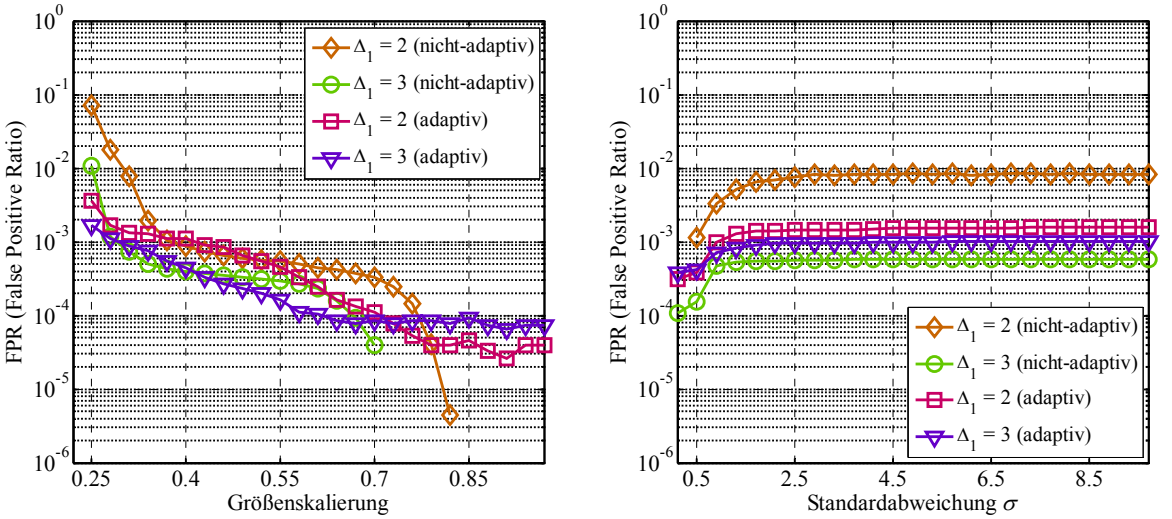


Abbildung E.21 Robustheit gegenüber Änderungen der Bildauflösung (links) und Bildschärfung (rechts)

Anhang F

Der Gaußsche Skalenraum

Für viele Anwendungen ist es hilfreich, bei der Untersuchung von digitalen Bildern sowohl feine als auch grobe Strukturen gleichzeitig zu erfassen. Zum Beispiel bei der Objekterkennung kann die Entfernung eines Objektes von der Kamera variieren. Dadurch verändert sich auch die Größe aller Details des Objektes, anhand derer das Objekt erkannt werden kann.

Eine Möglichkeit zur skalierungsunabhängigen Betrachtung von Bildern bietet der Gaußsche Skalenraum, der von Lindeberg [Lin98] eingehen untersucht wurde. Im Gaußschen Skalenraum ist es möglich, gleichzeitig grobe und feine Strukturen innerhalb eines Bildes zu erkennen. Als Basis dazu dienen Gauß-Masken $Gauß(x,y)$, die nach Gleichung (F.1) berechnet werden können.

$$Gauß(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{\left(x^2+y^2\right)}{2\sigma^2}} \tag{F.1}$$

Der Parameter σ dieser Gauß-Masken ist eine Variable, die den Gaußschen Skalenraum von $\sigma_{\min}$ bis $\sigma_{\max}$ aufspannt. Durch $\sigma_{\min}$ wird ein unterer Grenzwert zur Betrachtung sehr feiner Bilddetails definiert. Wird in einem zu untersuchenden Bild Rauschen erwartet, sollte dieser Parameter nicht zu klein gewählt werden, um den Einfluss des Rauschens zu begrenzen. Durch $\sigma_{\max}$ wird eine obere Grenze zur Betrachtung grober Bilddetails definiert.

Um ein Bild im Gaußschen Skalenraum darzustellen, muss es mit vielen Gauß-Masken von $\sigma_{\min}$ bis $\sigma_{\max}$ gefaltet werden. Dabei wird das Bild mit größer werdendem σ immer stärker geglättet, wobei feine Bilddetails immer mehr unterdrückt werden. Abbildung 5.5 (S. 76) zeigt schematisch den Gaußschen Skalenraum eines Beispielbildes.

In diesem Gaußschen Skalenraum können nun Berechnungen durchgeführt werden. Für eine Kantendetektierung können die einzelnen Bilder des Skalenraums zum Beispiel mit einer Gradientenfiltermaske $Grad(x,y)$ gefaltet werden. Da die Faltungsoperation dreier Operanden vertauschbar ist, kann die Faltung des geglätteten Bildes mit einer Gradientenfiltermaske durch eine Faltung des Originalbildes $I(x,y)$ mit dem Ergebnis der Faltung von $Gauß(x,y)$ und $Grad(x,y)$ ersetzt werden (siehe Gleichung (F.2)).

$$\begin{aligned} I_{Gauß_{grad}}(x,y) &= \left(I(x,y) * Gauß(x,y)\right) * Grad(x,y) \\ &= I(x,y) * \left(Gauß(x,y) * Grad(x,y)\right) \end{aligned} \tag{F.2}$$

Da die Faltung der Gauß-Maske $Gauß(x,y)$ mit der Gradientenfiltermaske $Grad(x,y)$ lediglich eine *n*-fache Ableitung der Gauß-Maske in x-Richtung ∂_x^n und eine *m*-fache Ableitung in y-Richtung ∂_y^m approximiert, kann diese Faltung auch direkt durch die analytische Ableitung der Formel zur Berechnung der Gauß-Maske (Gleichung (F.3)) ersetzt werden.

$$\begin{aligned} I_{Gauß_{n,m}}(x,y) &= I(x,y) * \partial_x^n \partial_y^m Gauß(x,y) \\ &= I(x,y) * Gauß_{n,m}(x,y) \end{aligned} \tag{F.3}$$

Damit die Ergebnisse der Berechnungen im Skalenraum skalierungsinvariant werden, müssen sie entsprechend dem Grad *n* der verwendeten Ableitungen in x- und Grad *m* in y-Richtung nach Gleichung (F.4) normiert werden (vgl. [Lin98]).

$$I_{Gauß_{n,m,norm}}(x,y) = \sigma^{n+m} \cdot I_{Gauß_{n,m}}(x,y) \tag{F.4}$$

Diese Normierung führt dazu, dass sich die Amplituden von $I_{Gauß_{n,m,norm}}(x,y)$ bei einer Vergrößerung oder Verkleinerung des Bildes nicht ändern, sondern sich nur hin zu einer anderen Skale σ des Skalenraums verschieben. Indem für jede Pixelposition (x,y) das Betragsmaximum im Skalenraum $I_{Gauß_{n,m,norm}}(x,y,\sigma)$ gesucht wird, kann für jeden Bildpunkt eine optimale Skale σ_{opt} bestimmt werden, unter der für diesen Bildpunkt die maximale Betragsamplitude im Skalenraum erreicht wird.

Bei einer Veränderung der Bildgröße verschiebt sich die optimale Skale σ_{opt} für jeden Bildpunkt nach Gleichung (F.4) linear mit dem Faktor der Größenänderung *S* hin zu $\sigma_{opt_{neu}}$.

$$\sigma_{opt_{neu}}(x \cdot S, y \cdot S) = \sigma_{opt}(x,y) \cdot S \tag{F.5}$$

Mit der beschriebenen Theorie des Gaußschen Skalenraums ist es möglich, beliebige auf Richtungsableitungen basierende Berechnungen innerhalb des Bildes skalierungsinvariant durchzuführen.

Die Theorie des Gaußschen Skalenraums findet in dieser Arbeit in Kapitel 5 bei der Entwicklung des Gray-Level-Blob-basierten Wasserzeichenverfahrens Anwendung.

Anhang G

Leistungsanalyse des Gray-Level-Blob-basierten Wasserzeichenverfahrens

Dieses Kapitel des Anhangs umfasst alle Untersuchungen zur Robustheit des Blob-basierten Wasserzeichenverfahrens aus Kapitel 5 gegenüber Störungen. In Abschnitt G.1 wird zunächst die Veränderung analysiert, die ein einzelner Gray-Level-Blob aufgrund von Störungen erfährt. Im Anschluss daran wird im Abschnitt G.2 die Robustheit der insgesamt eingebetteten Wasserzeichendaten ermittelt.

G.1 Auswirkungen von Störungen auf die Amplitude eines Blobs

Anhand einer Simulation wurde zunächst untersucht, in welchem Maß sich die Amplitude des größten Blobs B_0 eines Bildes infolge von Störungen (Angriffen) verändert. Die nachfolgenden Abbildungen zeigen die Ergebnisse der Simulation (Betrag der Differenz zwischen größter Amplitude B_0 vor und größter Amplitude $\hat{B}_0$ nach Anwendung der Störungen) gemittelt über die 52 verwendeten Testbilder. Dabei wurde der Betrag der größten Blob-Amplitude zuvor durch eine Grundverstärkung G_{base} vergrößert. Die Skalen wurden auf $\sigma_{\min} = 3$ und $\sigma_{\max} = 5$ festgelegt.

Die Kurven der linken Grafik einer jeden Abbildung zeigen die Ergebnisse der Simulation ohne Kontrastnormierung. Für die Werte der jeweils rechten Grafiken wurde die Differenz der Blob-Amplituden B_0 und $\hat{B}_0$ mithilfe der in Abschnitt 5.3.2 beschriebenen Normierungsfaktoren g und $\hat{g}$ wie folgt ermittelt:

$$\left| B_0 - \hat{B}_0 \right| = \hat{g} \cdot \left| B_0 \cdot g - \hat{B}_0 \cdot g \right| \tag{G.1}$$

Diese Normierung ist mit der in Abschnitt 5.3.2 beschriebenen Kontrastnormierung der Quantisierungsschrittweiten Δ und $\hat{\Delta}$ unmittelbar gleichzusetzen.

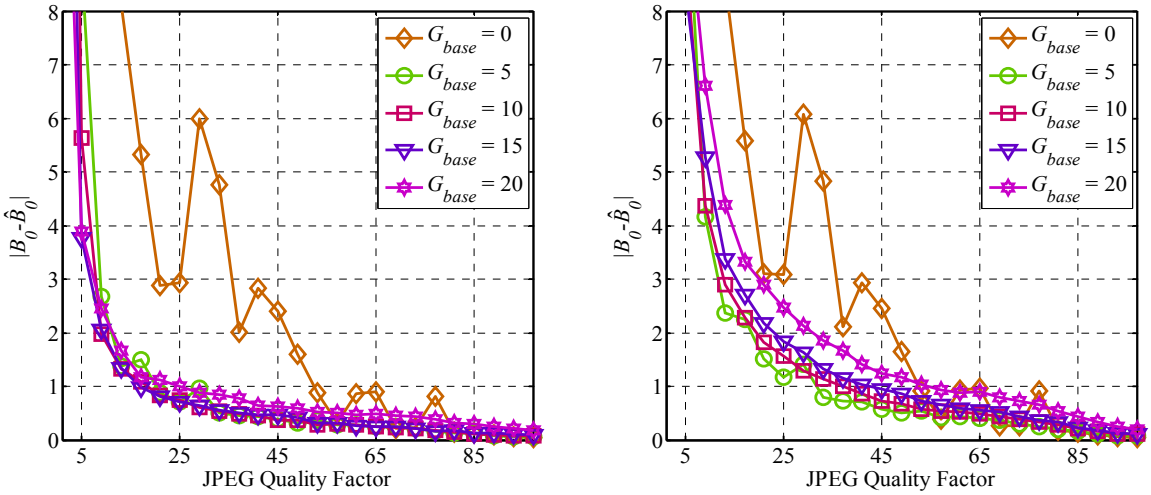


Abbildung G.1 Veränderung der Blob-Amplitude $|B_0|$ durch JPEG-Kompression ohne (links) und mit Kontrastnormierung (rechts)

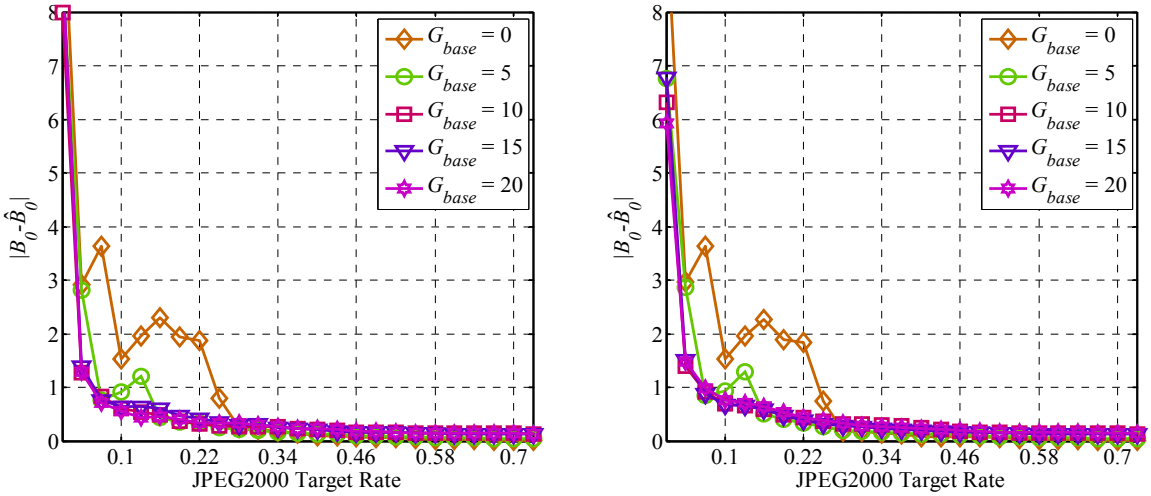


Abbildung G.2 Veränderung der Blob-Amplitude $|B_0|$ durch JPEG2000-Kompression ohne (links) und mit Kontrastnormierung (rechts)

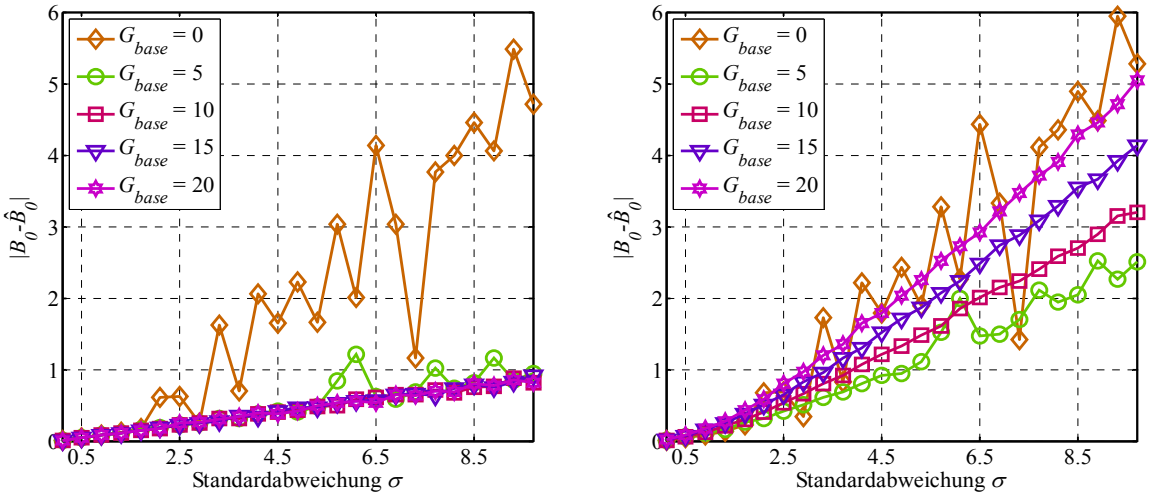


Abbildung G.3 Veränderung der Blob-Amplitude $|B_0|$ durch Gaußsches Rauschen ohne (links) und mit Kontrastnormierung (rechts)

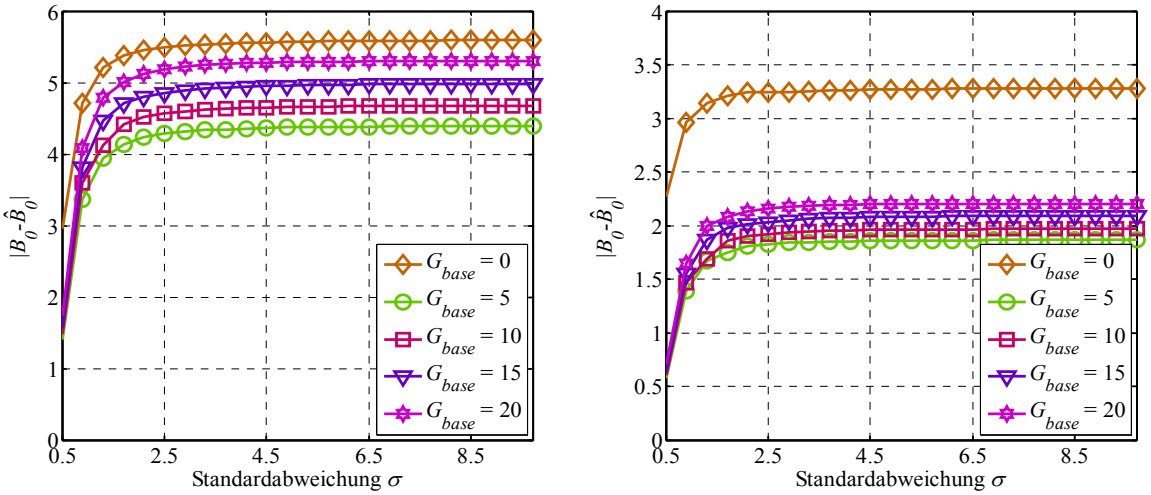


Abbildung G.4 Veränderung der Blob-Amplitude $|B_0|$ durch Gaußsche Tiefpassfilterung ohne (links) und mit Kontrastnormierung (rechts)

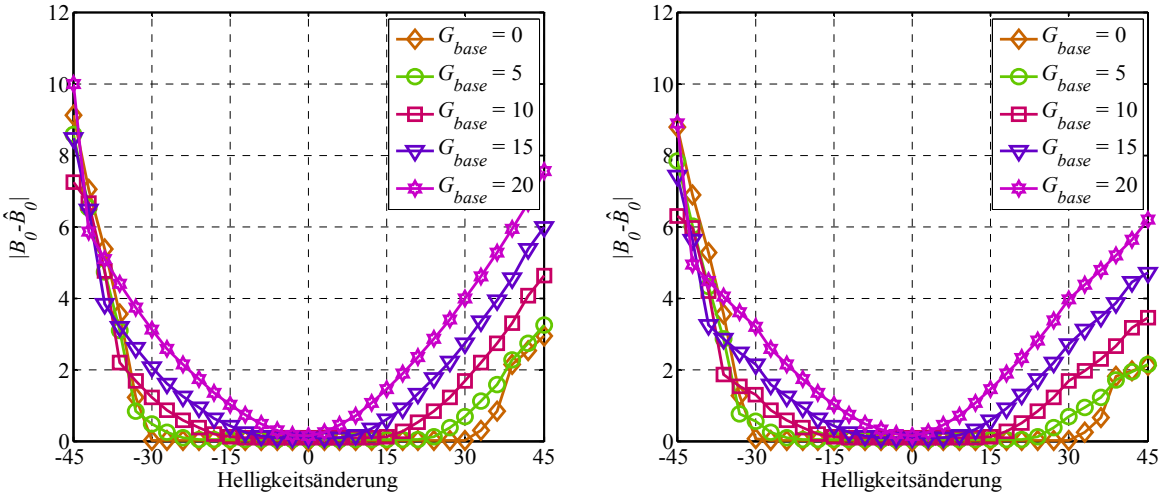


Abbildung G.5 Veränderung der Blob-Amplitude $|B_0|$ durch Helligkeitsänderungen ohne (links) und mit Kontrastnormierung (rechts)

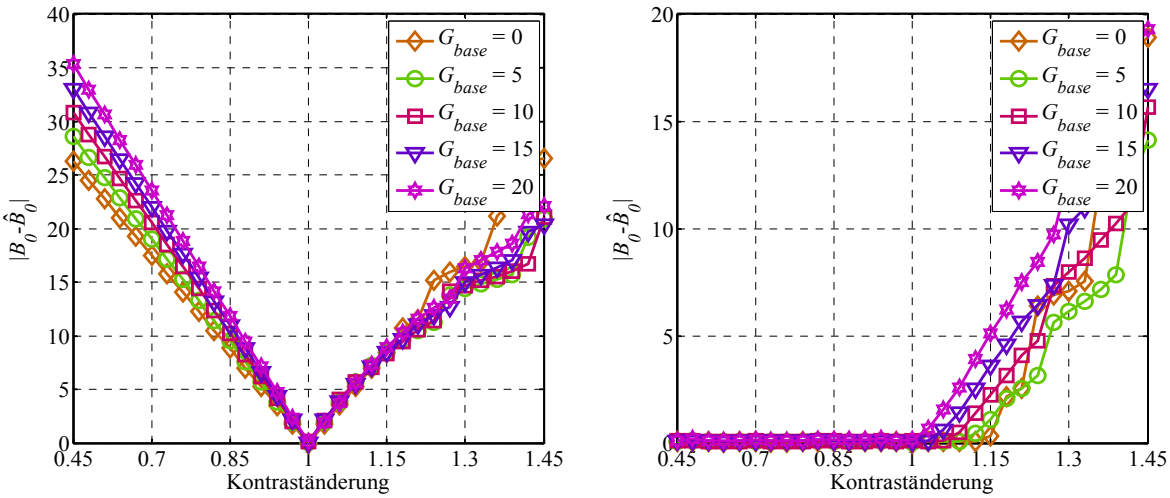


Abbildung G.6 Veränderung der Blob-Amplitude $|B_0|$ durch Kontraständerungen ohne (links) und mit Kontrastnormierung (rechts)

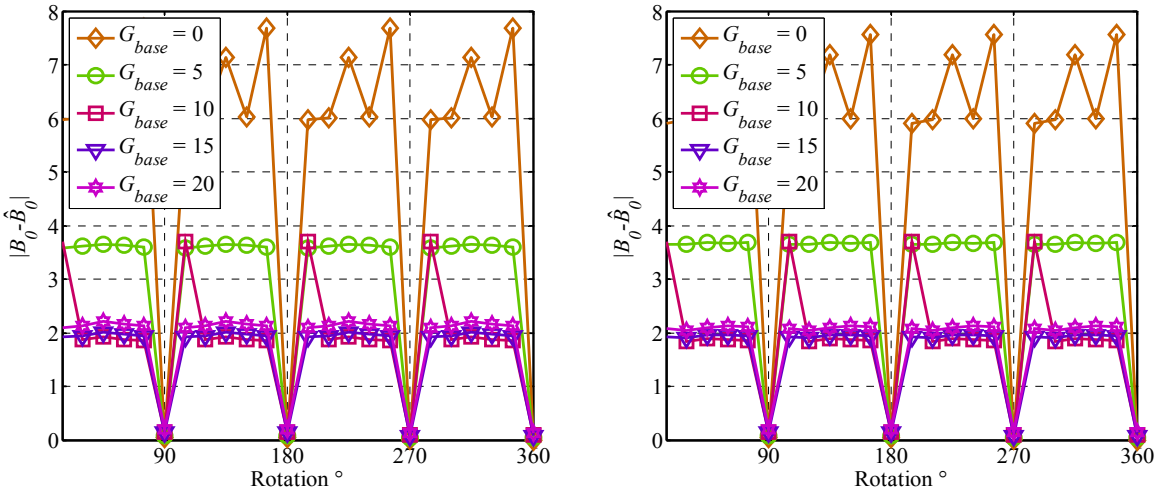


Abbildung G.7 Veränderung der Blob-Amplitude $|B_0|$ durch Rotation des Bildes ohne (links) und mit Kontrastnormierung (rechts)

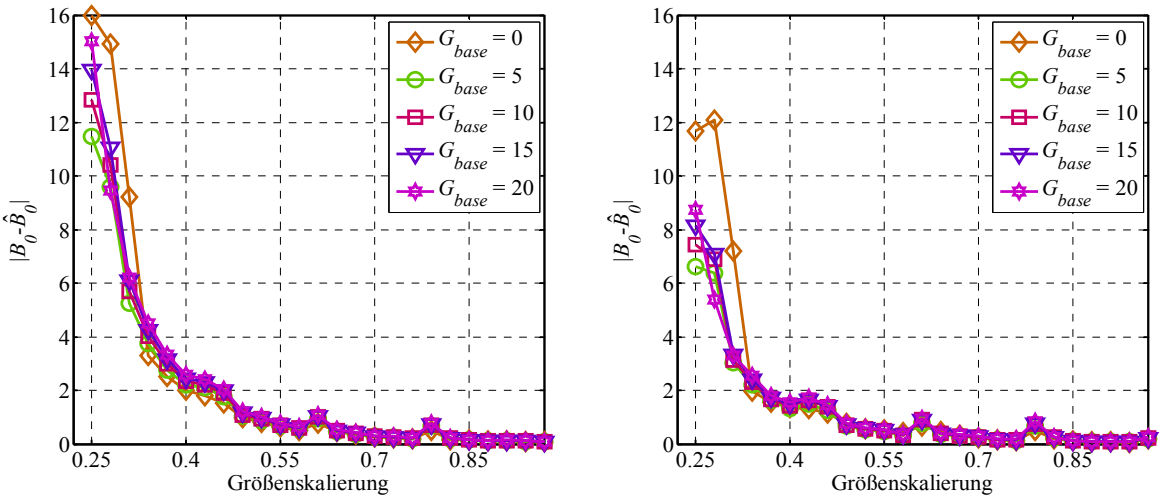


Abbildung G.8 Veränderung der Blob-Amplitude $|B_0|$ durch Änderung der Bildauflösung ohne (links) und mit Kontrastnormierung (rechts)

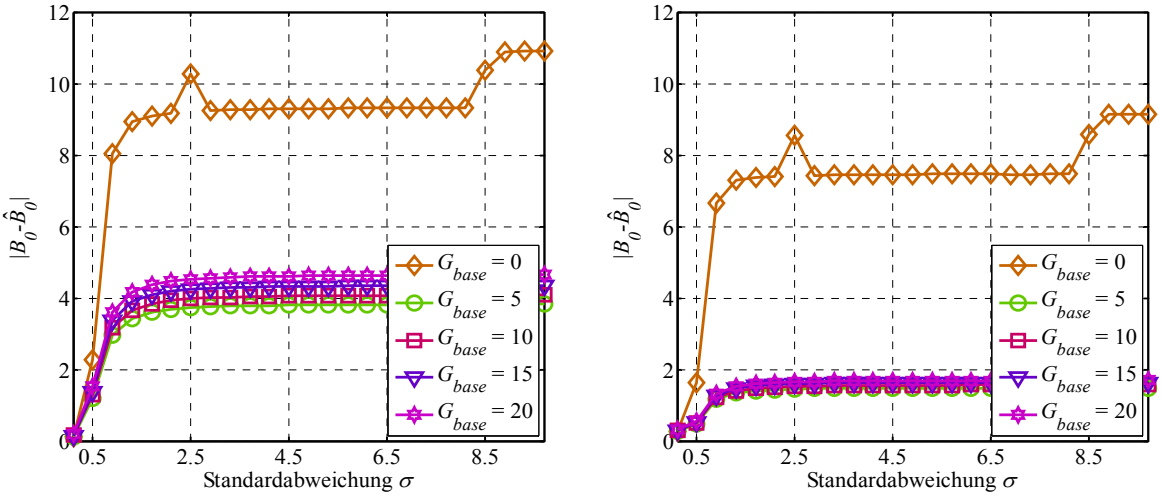


Abbildung G.9 Veränderung der Blob-Amplitude $|B_0|$ durch Unsharp-Mask-Filterung ohne (links) und mit Kontrastnormierung (rechts)

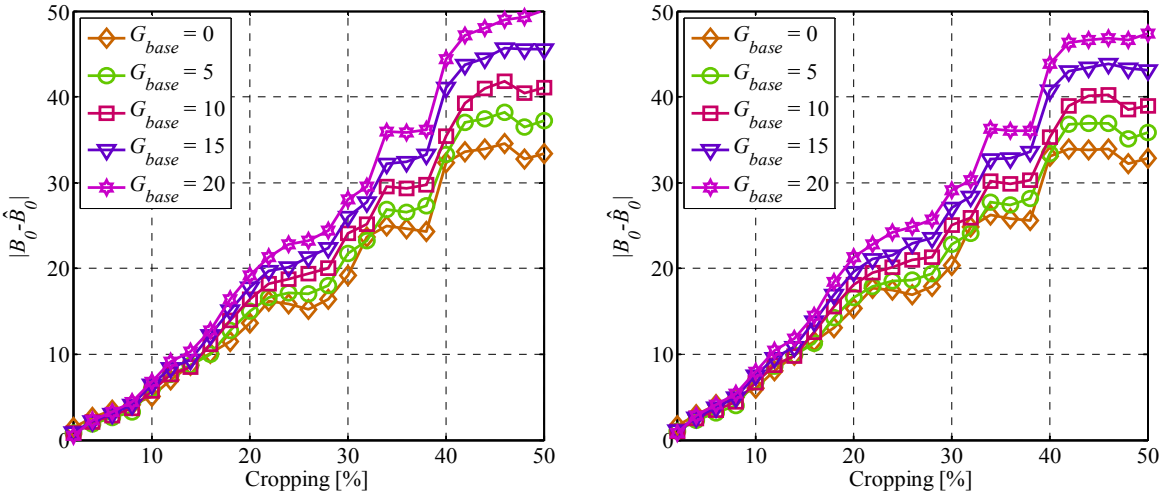


Abbildung G.10 Veränderung der Blob-Amplitude $|B_0|$ durch Cropping von Bildzeilen ohne (links) und mit Kontrastnormierung (rechts)

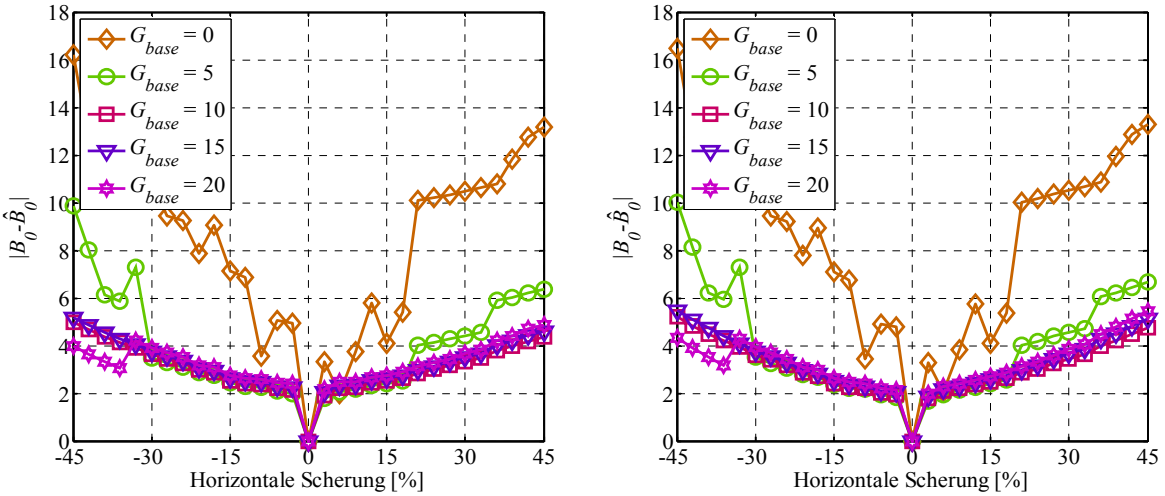


Abbildung G.11 Veränderung der Blob-Amplitude $|B_0|$ durch horizontale Scherung des Bildes ohne (links) und mit Kontrastnormierung (rechts)

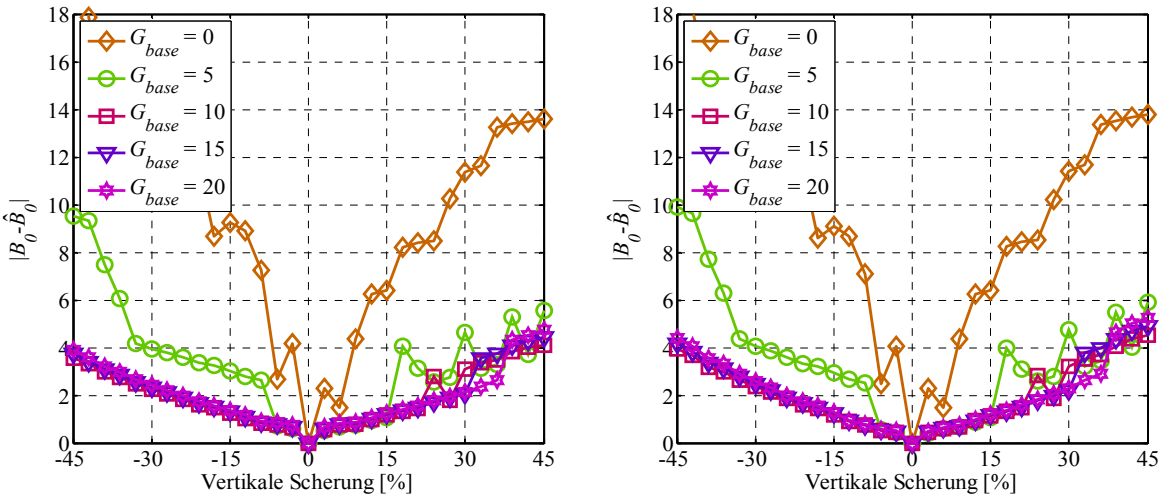


Abbildung G.12 Veränderung der Blob-Amplitude $|B_0|$ durch vertikale Scherung des Bildes ohne (links) und mit Kontrastnormierung (rechts)

G.2 Robustheit der eingebetteten Wasserzeichendaten

Um die Robustheit der insgesamt durch das Blob-basierte Wasserzeichenverfahren eingebetteten Daten zu bestimmen, wurden in jedem der 52 Testbilder, wie in Abschnitt 5.3.2 beschrieben, die $K+1=33$ Gray-Level-Blobs mit den größten *LOG*-Amplitudenwerten ausgewählt. Nach einer Grundverstärkung G_{base} sowie einer Quantisierung der Blob-Amplituden (Einbetten der Datenbits) wurde im Rahmen einer Simulation die Abweichung der jeweiligen Blob-Amplitudenwerte als Folge der Anwendung von Störungen berechnet.

Die nachfolgend dargestellten Abbildungen zeigen die ermittelten Bitfehlerverhältnisse dieser Simulation. Die Ergebnisse werden vor allem infolge geometrischer Bildveränderungen stark durch Desynchronisation beeinflusst, wenn aufgrund von Blob-Überlappungen nicht dieselben 33 Blobs gefunden werden (siehe dazu Kapitel 6).

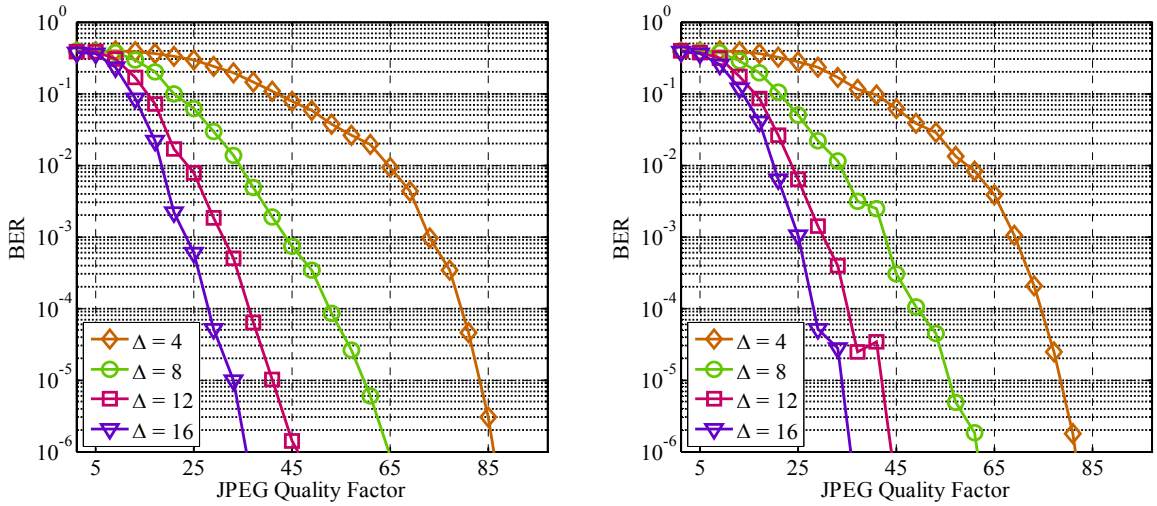


Abbildung G.13 Robustheit gegenüber JPEG-Kompression mit den Simulationsparametern: $\sigma_{\min}=3$, $\sigma_{\max}=5$ sowie $G_{base}=5$ (links) und $G_{base}=10$ (rechts)

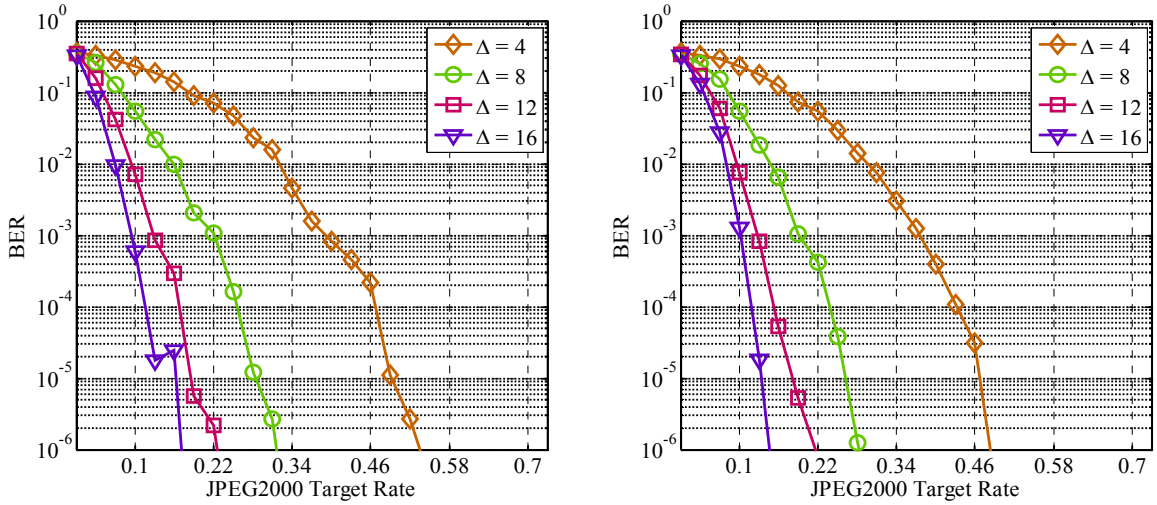


Abbildung G.14 Robustheit gegenüber JPEG2000-Kompression mit den Simulationsparametern: $\sigma_{\min}=3$, $\sigma_{\max}=5$ sowie $G_{base}=5$ (links) und $G_{base}=10$ (rechts)

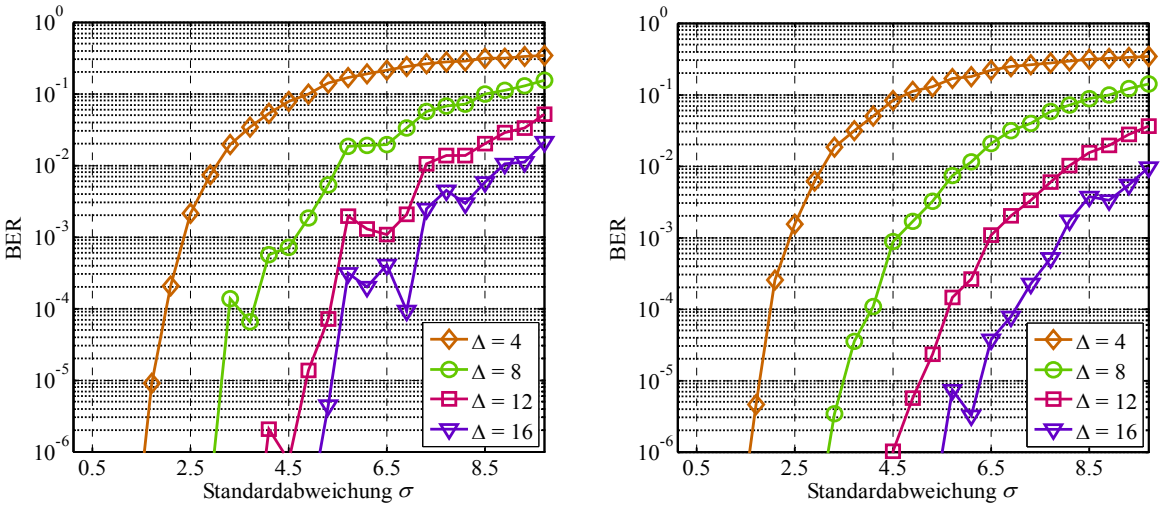


Abbildung G.15 Robustheit gegenüber Gaußschem Rauschen mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

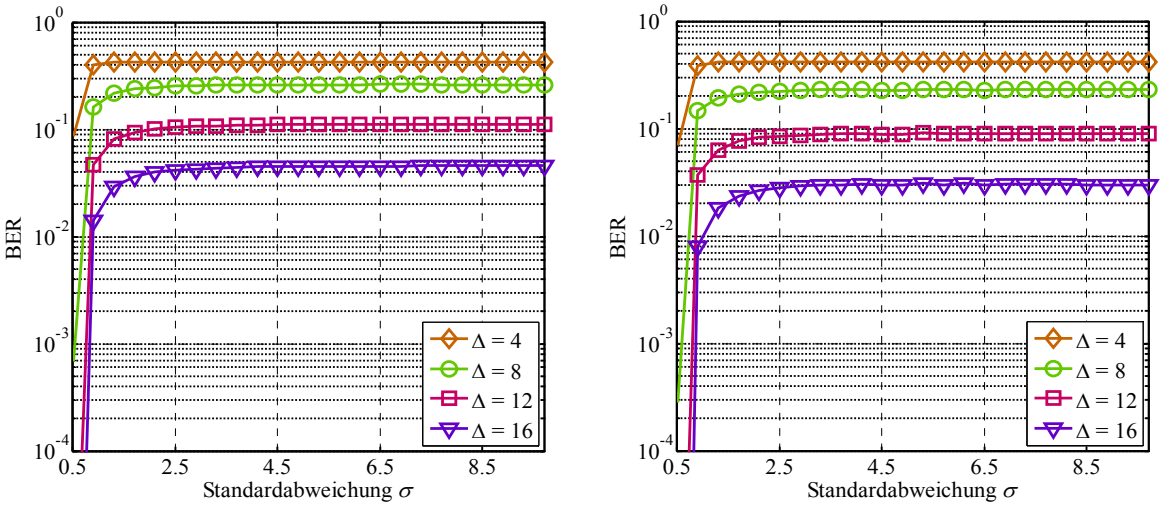


Abbildung G.16 Robustheit gegenüber Gaußscher Tiefpassfilterung mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

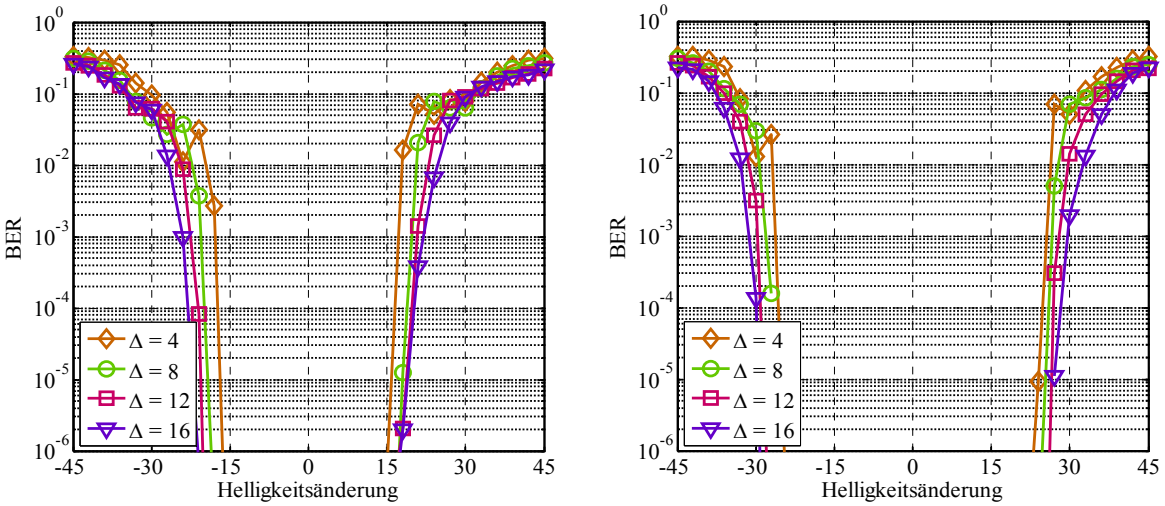


Abbildung G.17 Robustheit gegenüber Helligkeitsänderungen mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

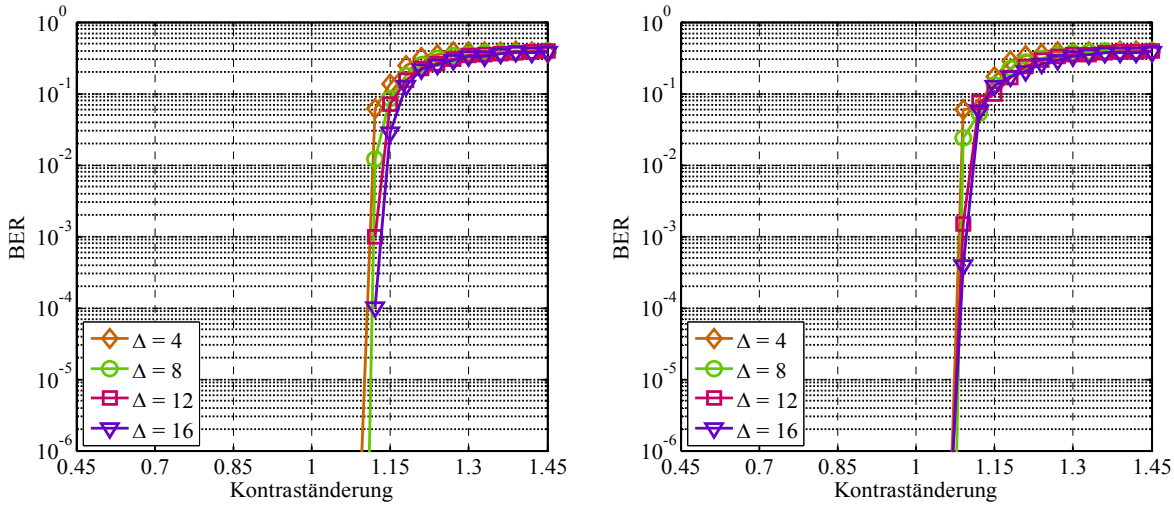


Abbildung G.18 Robustheit gegenüber Kontraständerungen mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

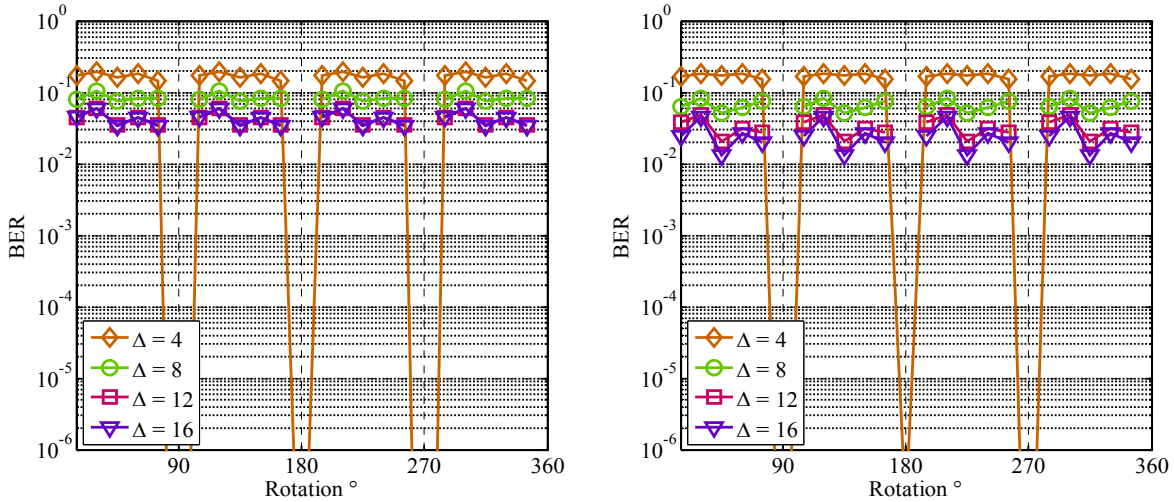


Abbildung G.19 Robustheit gegenüber Rotation des Bildes mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

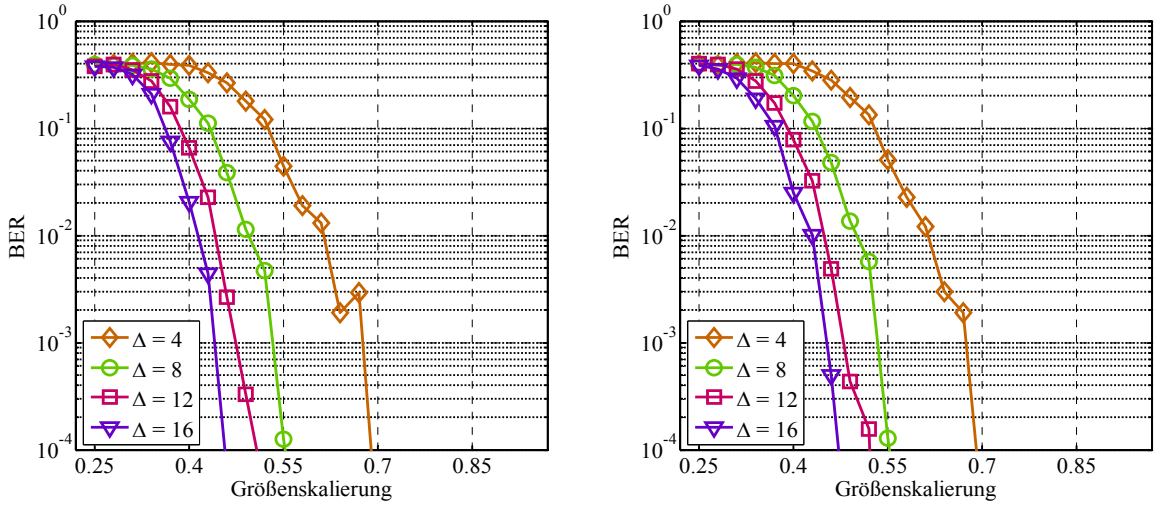


Abbildung G.20 Robustheit gegenüber Änderung der Bildauflösung mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

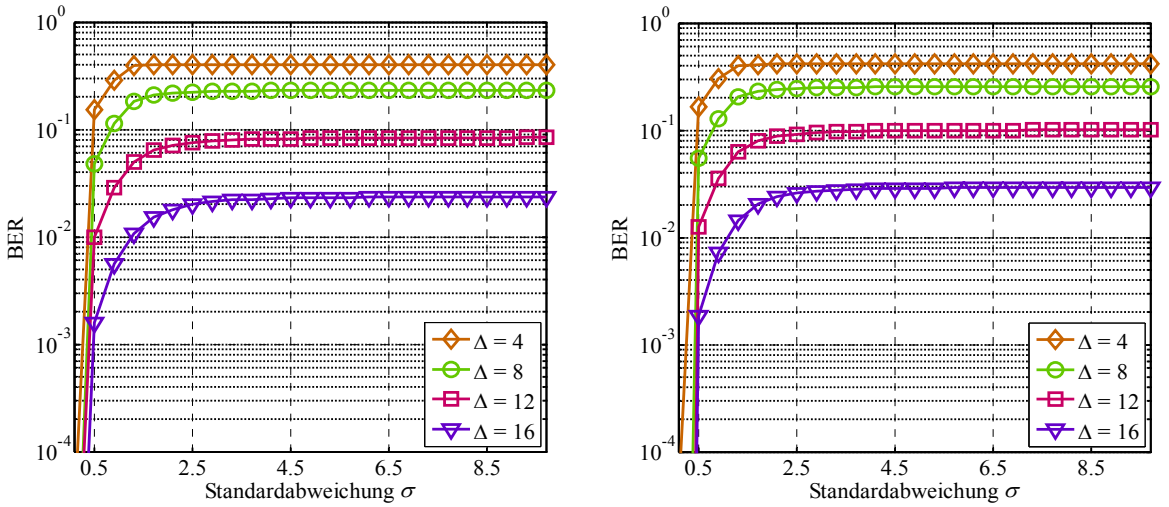


Abbildung G.21 Robustheit gegenüber Unsharp-Mask-Filterung mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

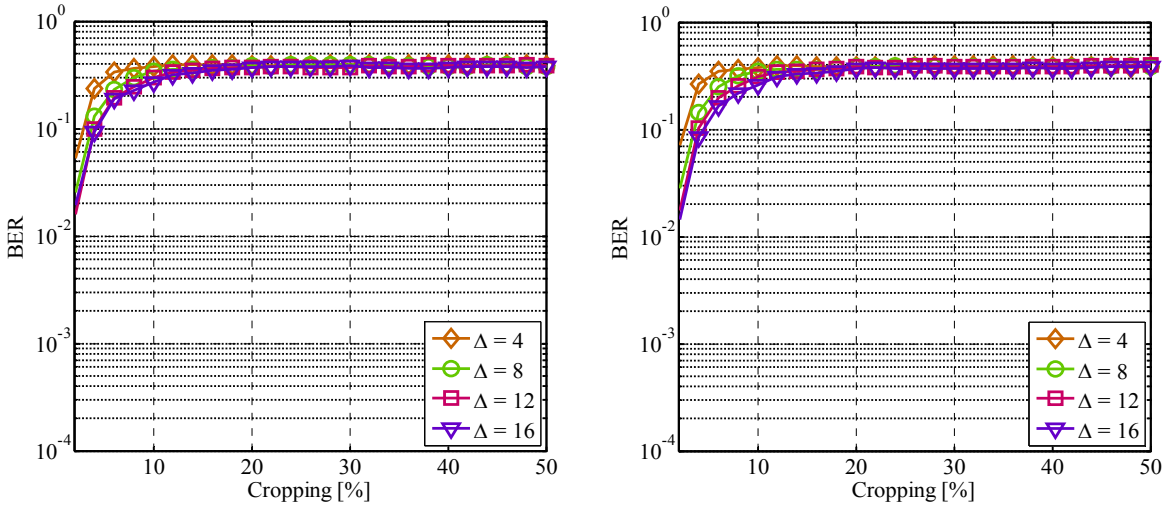


Abbildung G.22 Robustheit gegenüber Cropping von Bildzeilen mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

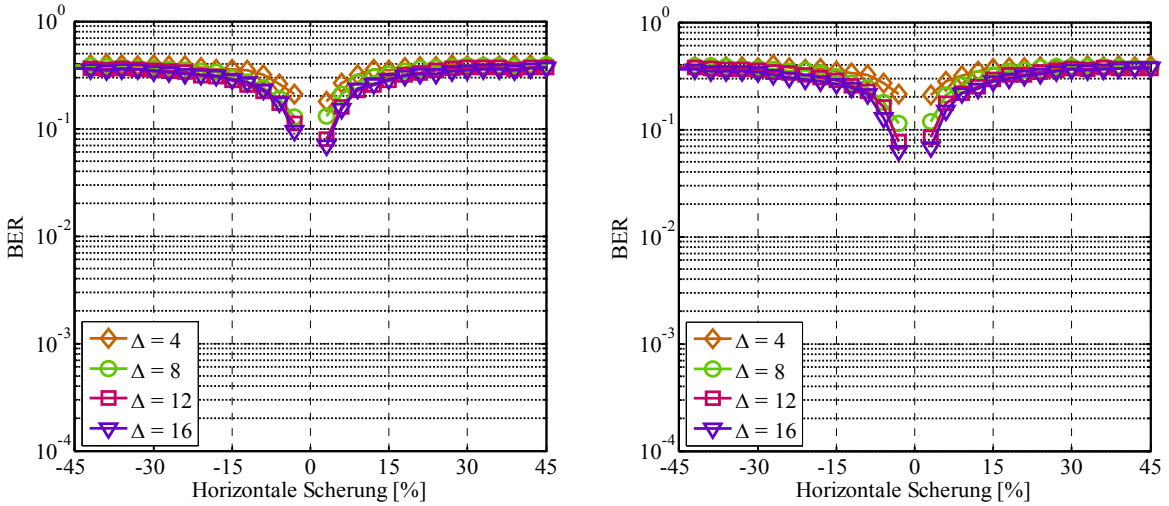


Abbildung G.23 Robustheit gegenüber horizontaler Scherung des Bildes mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

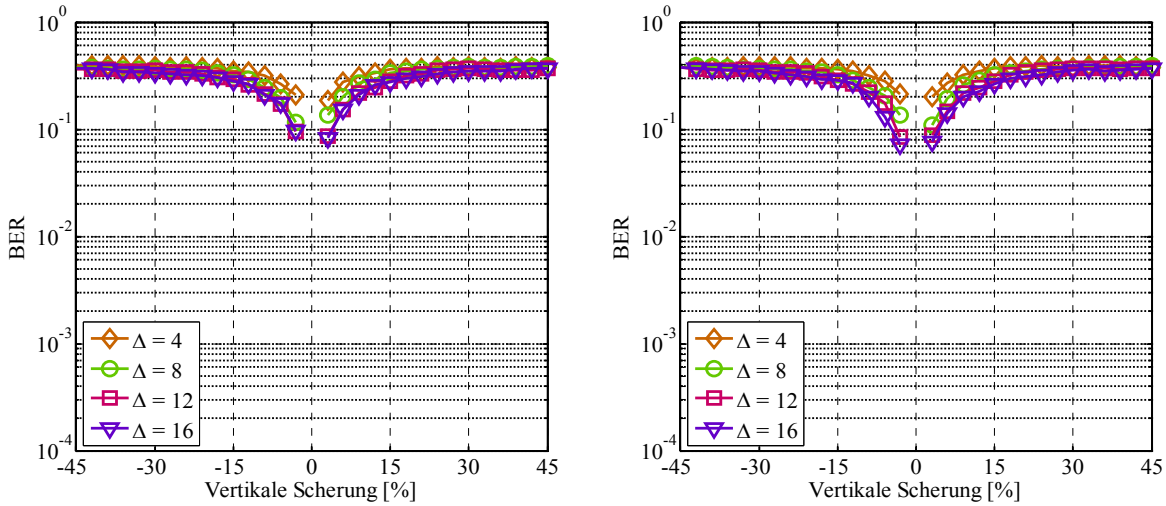


Abbildung G.24 Robustheit gegenüber vertikaler Scherung des Bildes mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

Anhang H

Leistungsanalyse des erweiterten Blob-basierten Wasserzeichenverfahrens

In Kapitel 6 wurde ein Verfahren entwickelt, mit dem neben gewöhnlichen Sustitutionsfehlern (binär: 0→1 bzw. 1→0) auch Einfügungen und Auslöschungen erkannt und korrigiert werden können. Die Leistungsfähigkeit dieses neuen Verfahrens wird in Abschnitt H.1 untersucht. In Abschnitt H.2 werden die Simulationsergebnisse des in Kapitel 5 entwickelten, um die beschriebene Resynchronisation erweiterten, Gray-Level-Blob-basierten Wasserzeichenverfahrens dargestellt.

H.1 Leistungsfähigkeit der entwickelten Resynchronisation

Die folgenden Abbildungen zeigen die mittels Simulation berechneten Bitfehlerverhältnisse der in Abschnitt 6.3 entwickelten Resynchronisationsverfahren “MSTrellis1 - MSTrellis3”. Für einen Leistungsvergleich sind ebenso die Simulationsergebnisse für einen nicht-modifizierten Viterbi-Decoder “no-resync” dargestellt. Die Berechnungen wurden für unterschiedliche Nachrichtenlängen L und Auftretenswahrscheinlichkeiten für Einfügungen/Auslöschungen P_d durchgeführt. Die Wahrscheinlichkeit für Substitutionsfehler folgt Gleichung (6.5).

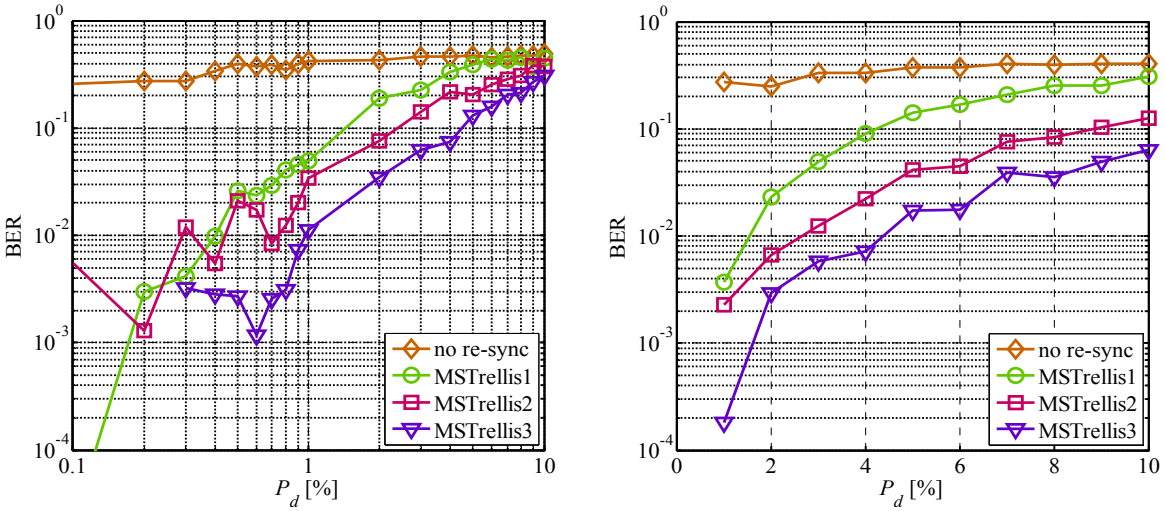


Abbildung H.1 Bitfehlerverhältnis nach Resynchronisation von Einfügungen/Auslöschungen bei einer Nachrichtenlänge von L = 500 Bit (links) bzw. L = 50 Bit (rechts) mit Coderate R_c = 1/2 und einem E_b/N_0 = 40db

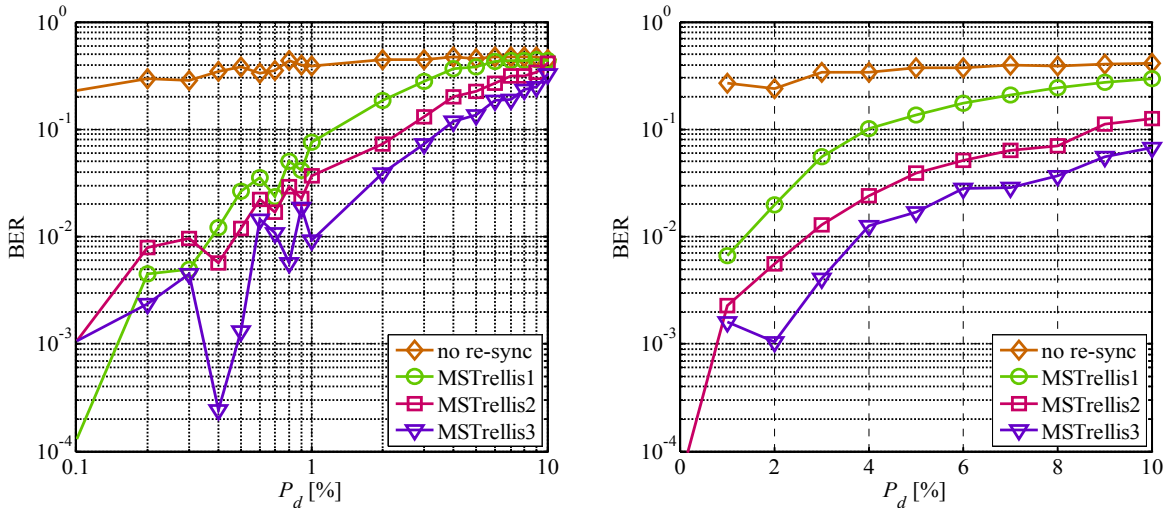


Abbildung H.2 Bitfehlerverhältnis nach Resynchronisation von Einfügungen/Auslöschungen bei einer Nachrichtenlänge von L = 500 Bit (links) bzw. L = 50 Bit (rechts) mit Coderate R_c = 1/2 und einem E_b/N_0 = 35db

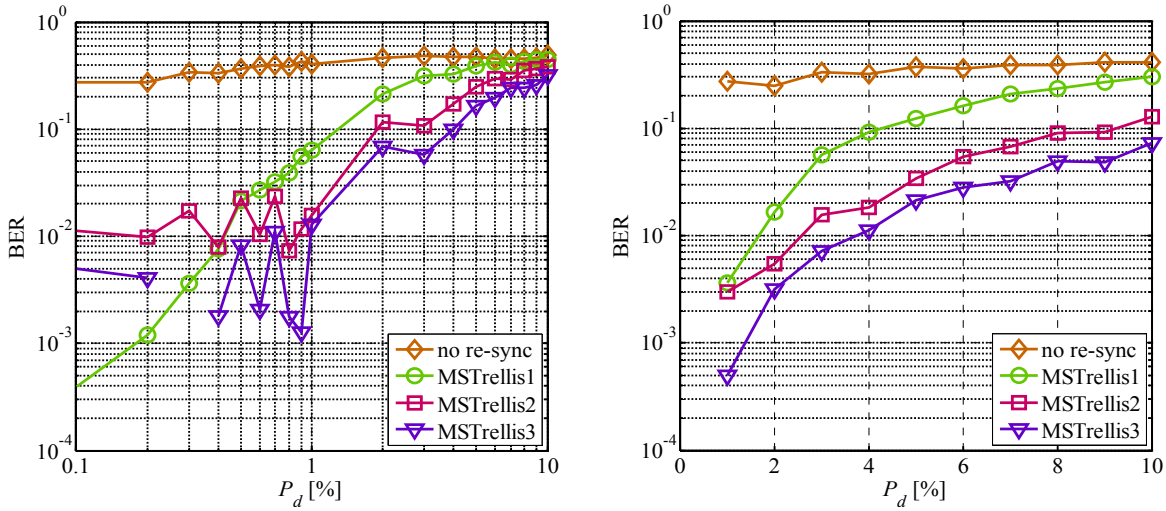


Abbildung H.3 Bitfehlerverhältnis nach Resynchronisation von Einfügungen/Auslöschungen bei einer Nachrichtenlänge von L = 500 Bit (links) bzw. L = 50 Bit (rechts) mit Coderate R_c = 1/2 und einem E_b/N_0 = 30db

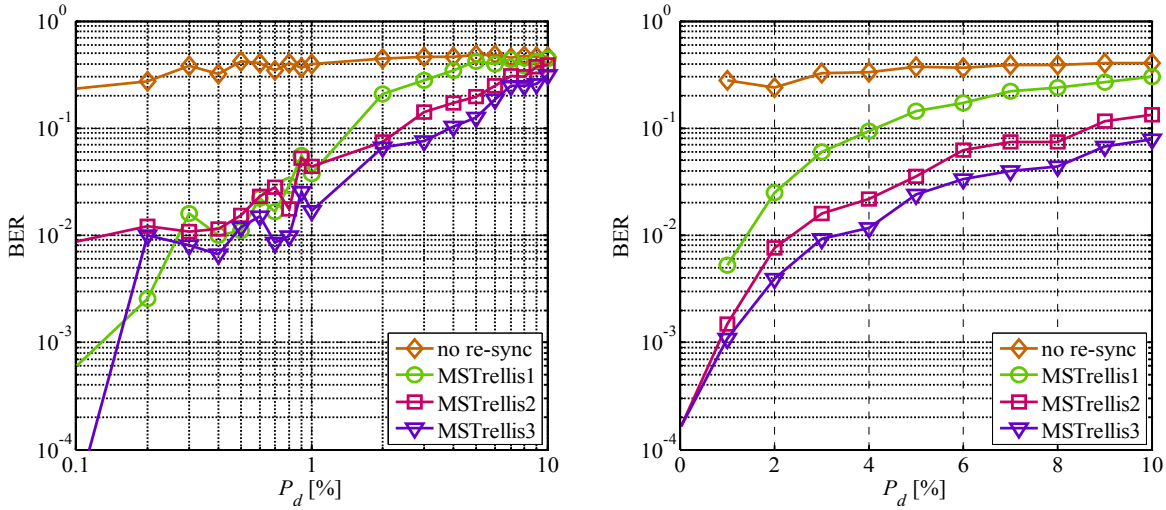


Abbildung H.4 Bitfehlerverhältnis nach Resynchronisation von Einfügungen/Auslöschungen bei einer Nachrichtenlänge von L = 500 Bit (links) bzw. L = 50 Bit (rechts) mit Coderate R_c = 1/2 und einem E_b/N_0 = 25db

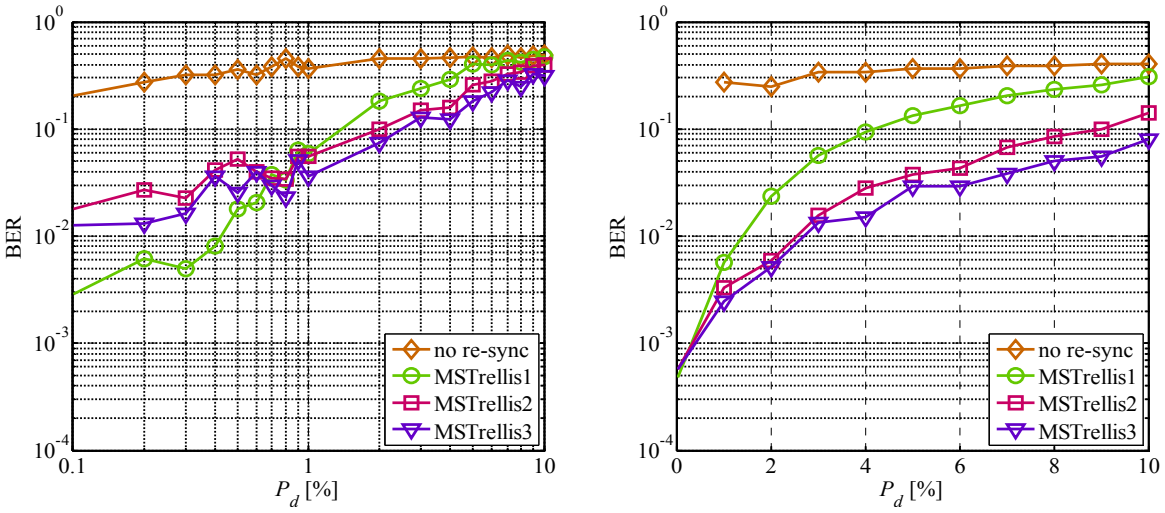


Abbildung H.5 Bitfehlerverhältnis nach Resynchronisation von Einfügungen/Auslöschungen bei einer Nachrichtenlänge von L = 500 Bit (links) bzw. L = 50 Bit (rechts) mit Coderate R_c = 1/2 und einem E_b/N_0 = 20db

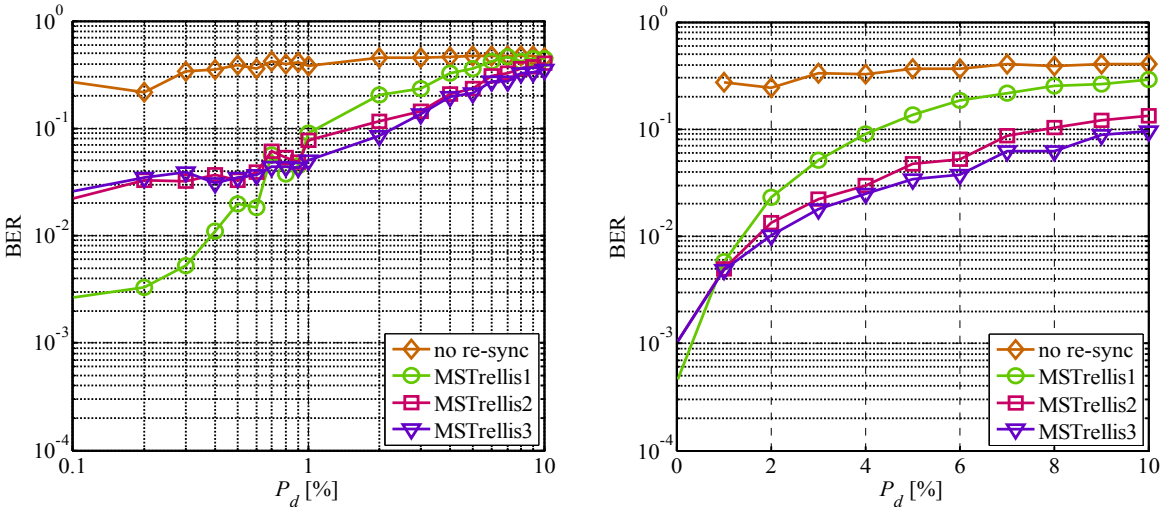


Abbildung H.6 Bitfehlerverhältnis nach Resynchronisation von Einfügungen/Auslöschungen bei einer Nachrichtenlänge von L = 500 Bit (links) bzw. L = 50 Bit (rechts) mit Coderate R_c = 1/2 und einem E_b/N_0 = 15db

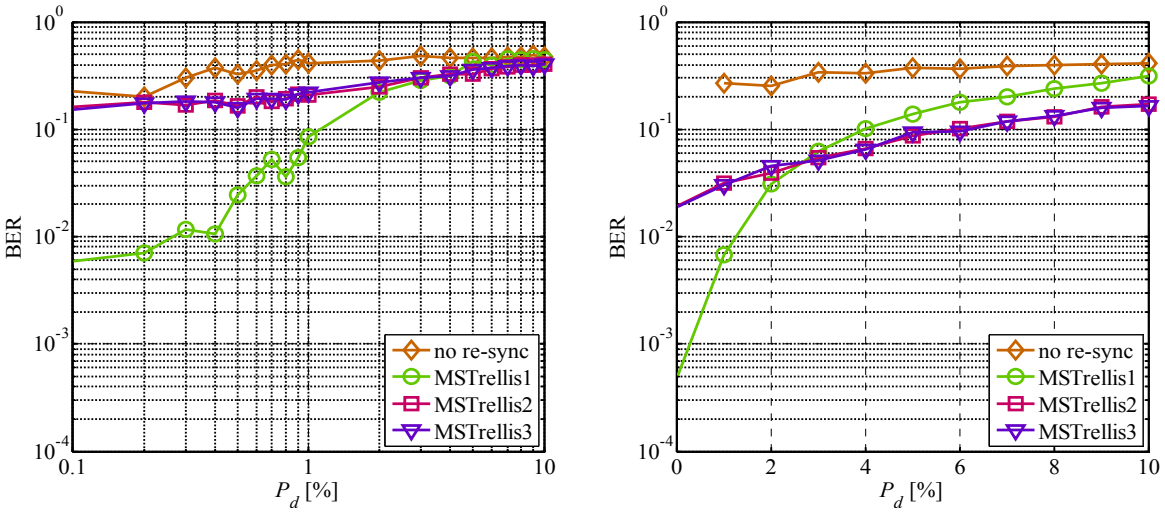


Abbildung H.7 Bitfehlerverhältnis nach Resynchronisation von Einfügungen/Auslöschungen bei einer Nachrichtenlänge von L = 500 Bit (links) bzw. L = 50 Bit (rechts) mit Coderate R_c = 1/2 und einem E_b/N_0 = 10db

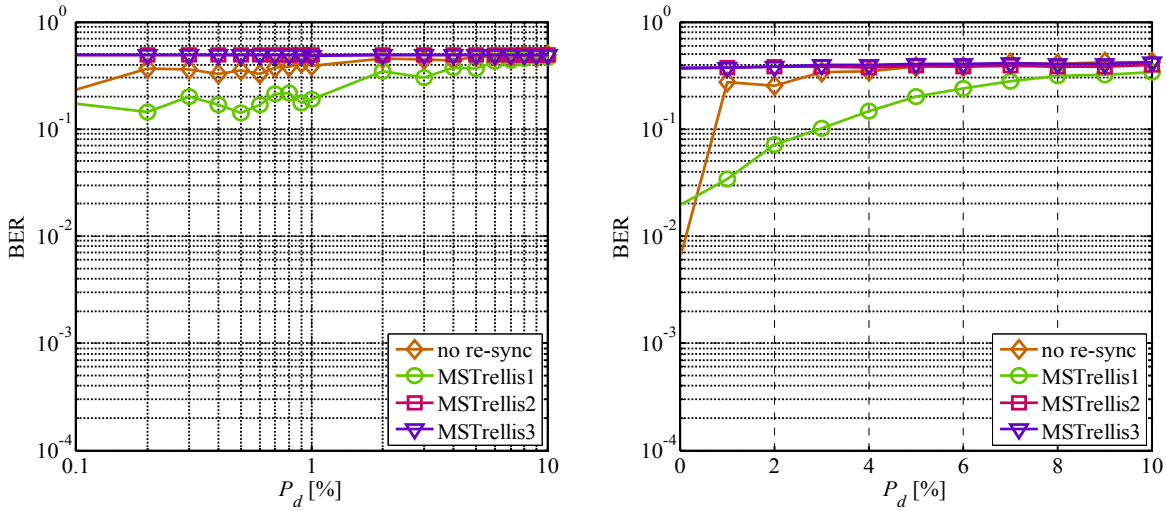


Abbildung H.8 Bitfehlerverhältnis nach Resynchronisation von Einfügungen/Auslöschungen bei einer Nachrichtenlänge von $L = 500$ Bit (links) bzw. $L = 50$ Bit (rechts) mit Coderate $R_c = 1/2$ und einem $E_b/N_0 = 5$db

H.2 Robustheit der eingebetteten Wasserzeichendaten

In den nachfolgenden Abbildungen sind die Ergebnisse der Robustheitssimulationen für das erweiterte Blob-basierte Wasserzeichenverfahren dargestellt.

Das in Kapitel 5 entwickelte und, wie in Kapitel 6 beschrieben, um eine Resynchronisation während der Fehlerkorrektur ergänzte Verfahren wurde unter den gleichen Simulationsbedingungen getestet wie das Verfahren ohne Resynchronisation (siehe Anhang G).

In jedem der 52 Testbilder wurden die $K+1=33$ Gray-Level-Blobs mit den größten *LOG*-Amplitudenwerten ausgewählt. Nach einer Grundverstärkung G_{base} sowie einer Quantisierung der Blob-Amplituden (Einbetten der Datenbits) wurde die Abweichung der jeweiligen Blob-Amplitudenwerte als Folge der Anwendung von Störungen berechnet.

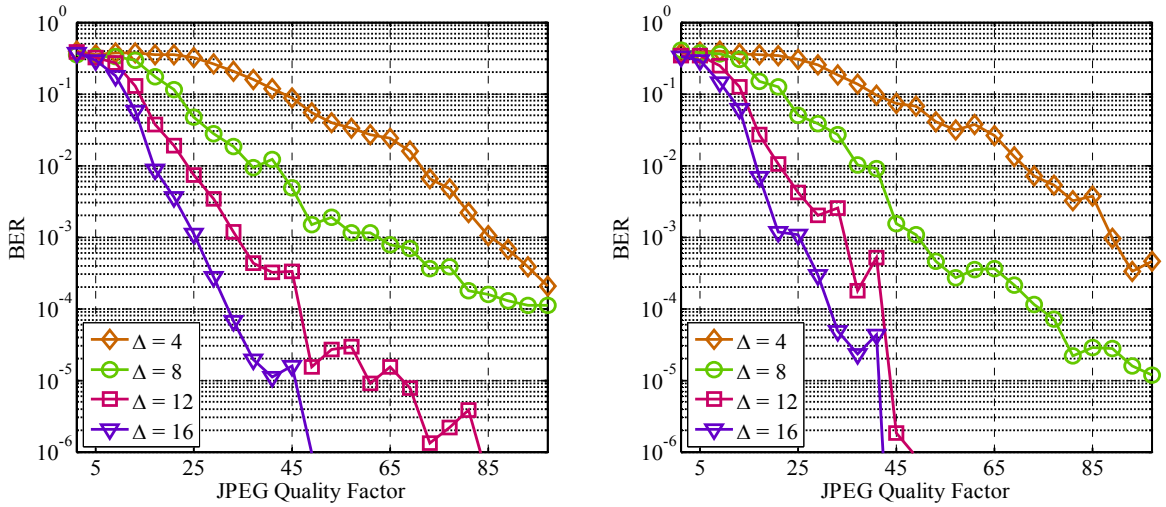


Abbildung H.9 Robustheit gegenüber JPEG-Kompression mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

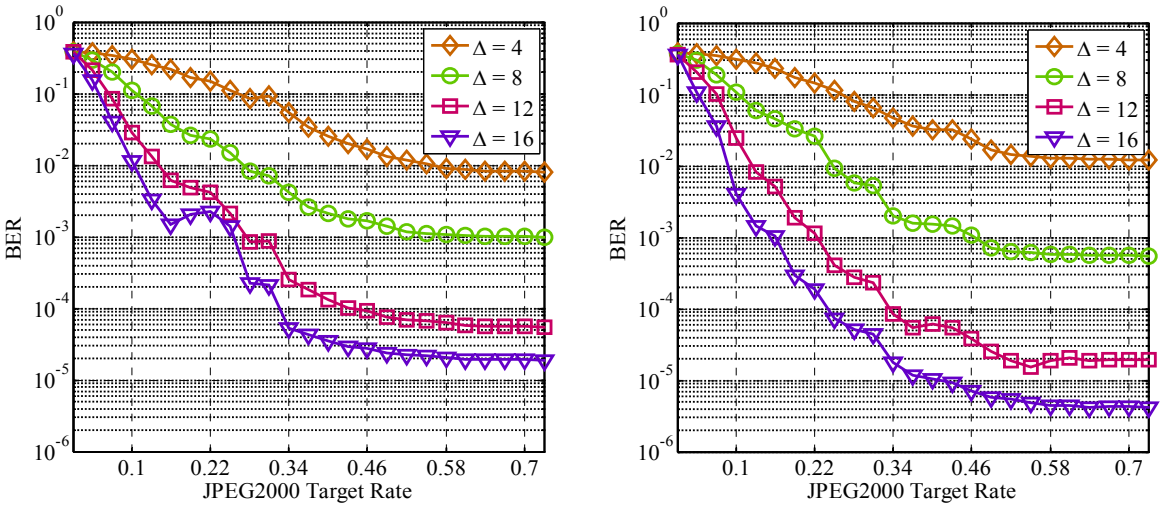


Abbildung H.10 Robustheit gegenüber JPEG2000-Kompression mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

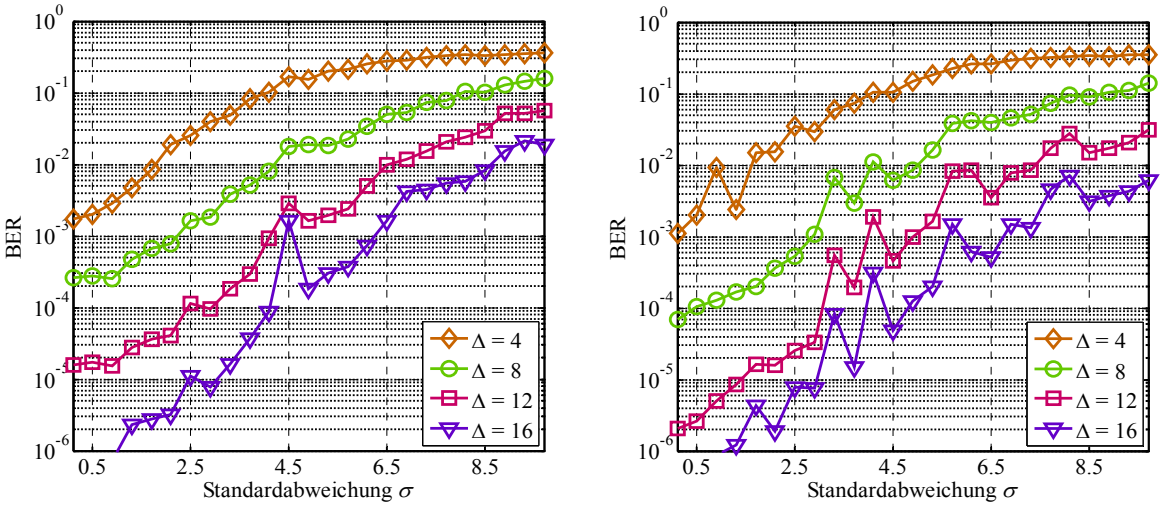


Abbildung H.11 Robustheit gegenüber Gaußschem Rauschen mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

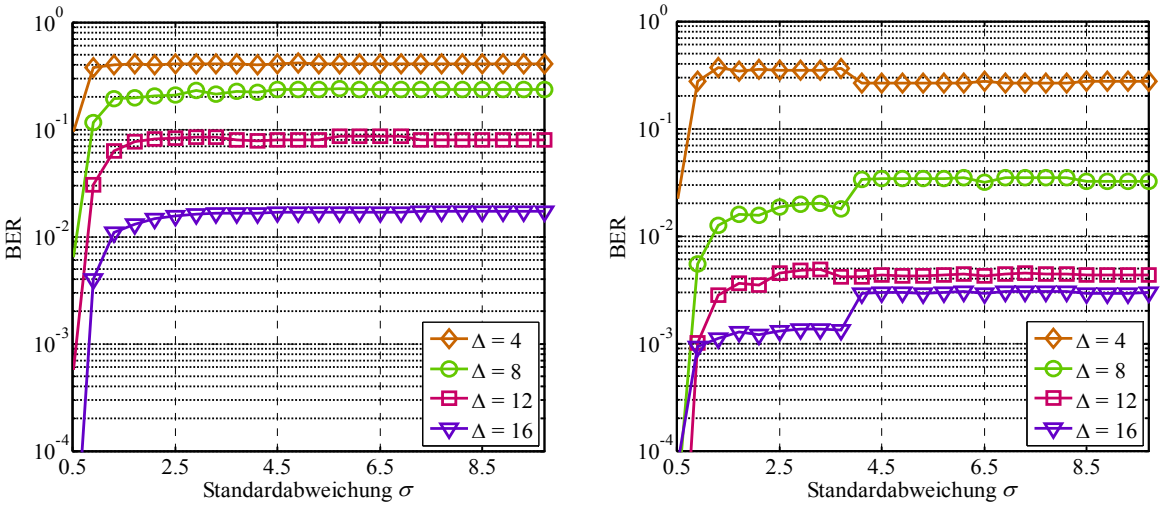


Abbildung H.12 Robustheit gegenüber Gaußscher Tiefpassfilterung mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

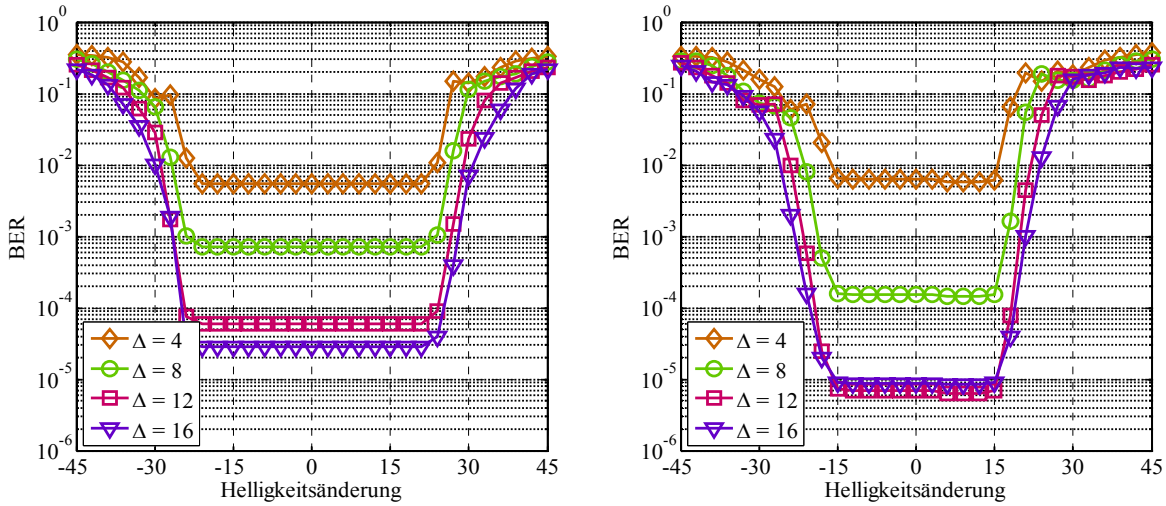


Abbildung H.13 Robustheit gegenüber Helligkeitsänderungen mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

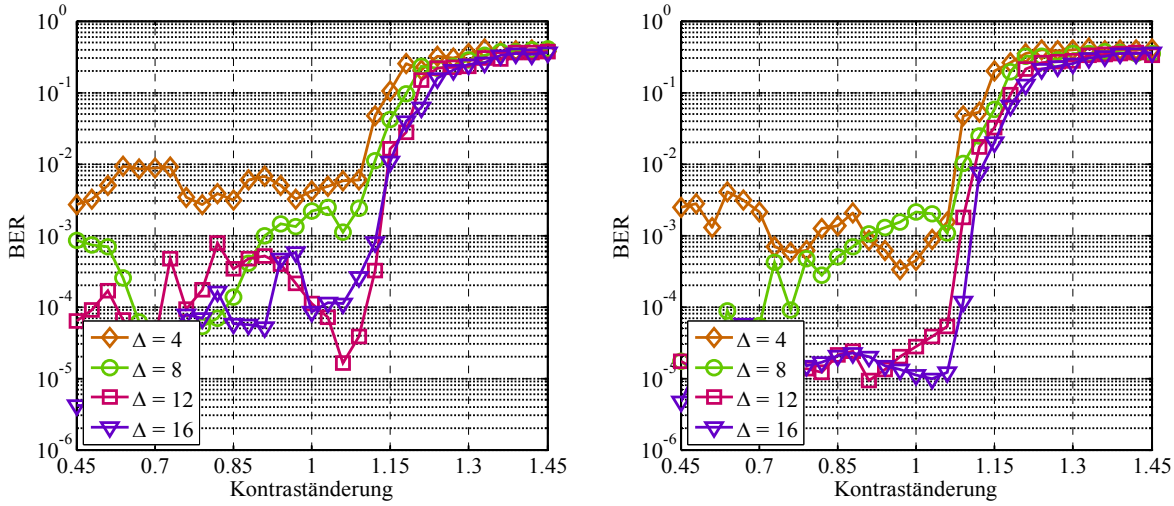


Abbildung H.14 Robustheit gegenüber Kontraständerungen mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

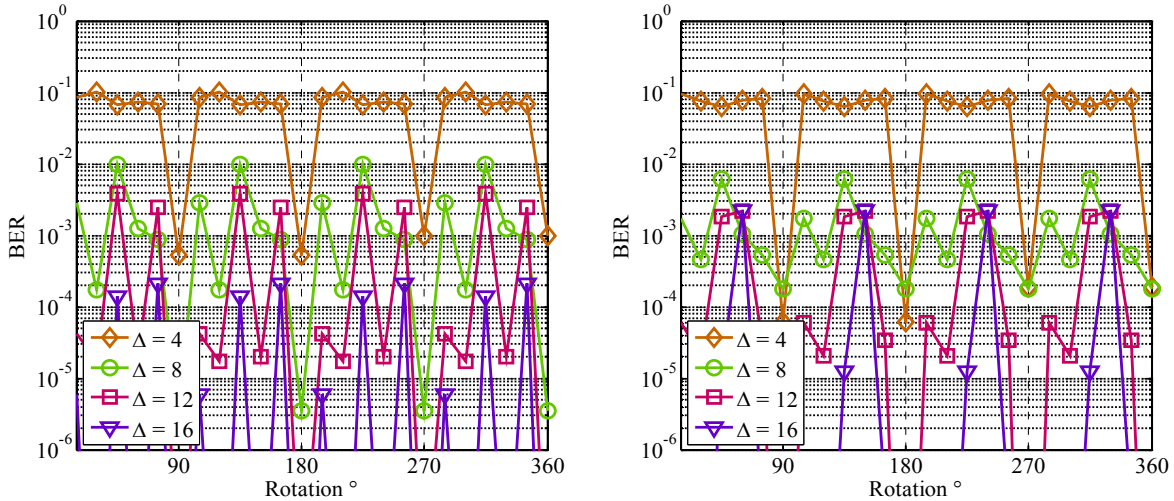


Abbildung H.15 Robustheit gegenüber Rotation des Bildes mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

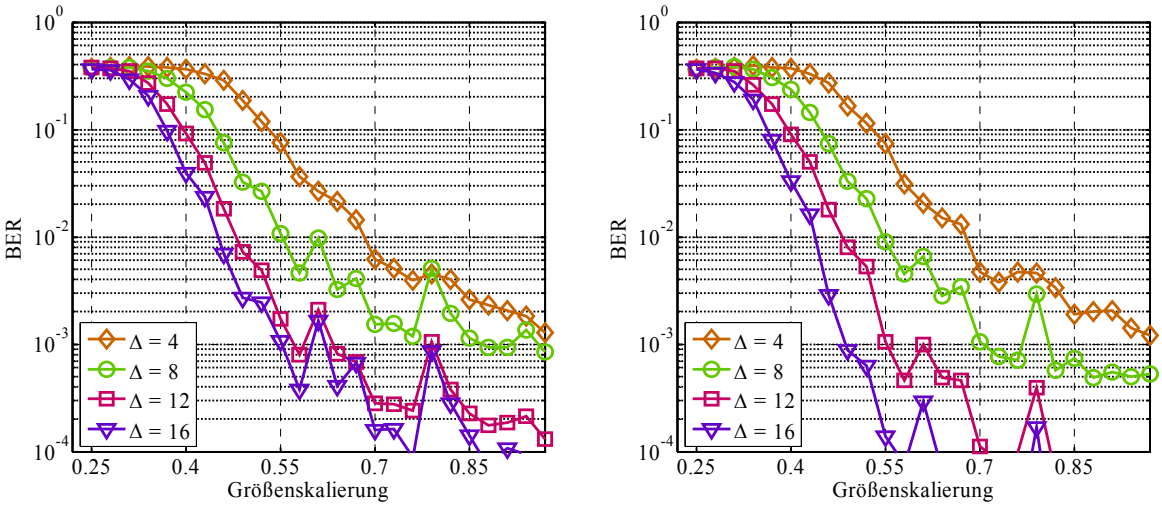


Abbildung H.16 Robustheit gegenüber Änderung der Bildauflösung mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

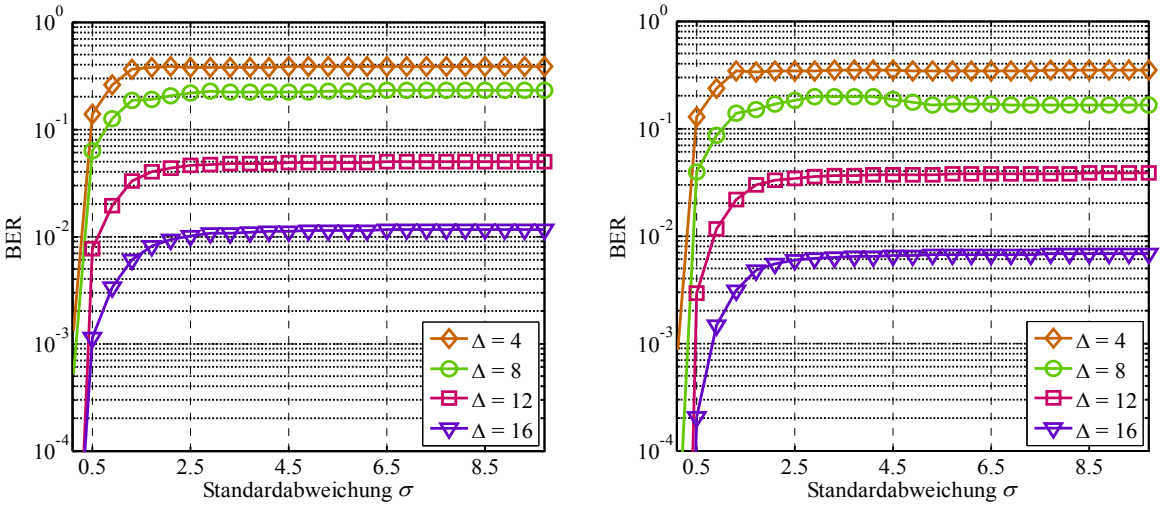


Abbildung H.17 Robustheit gegenüber Unsharp-Mask-Filterung mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

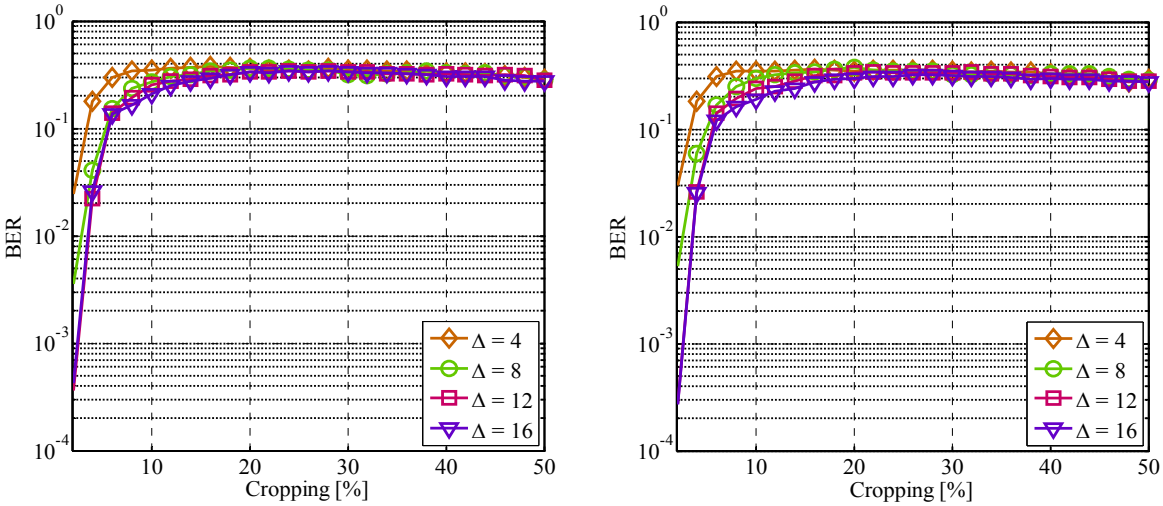


Abbildung H.18 Robustheit gegenüber Cropping von Bildzeilen mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

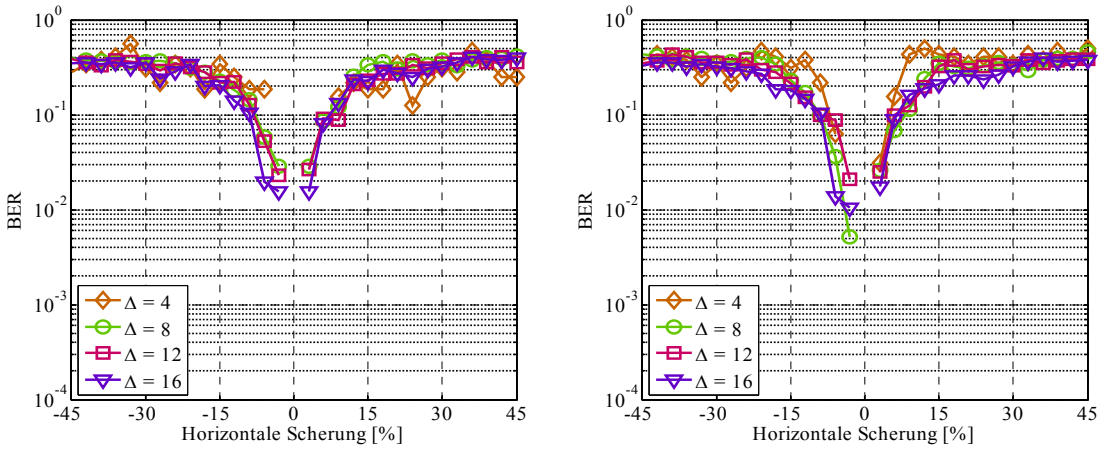


Abbildung H.19 Robustheit gegenüber horizontaler Scherung des Bildes mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)

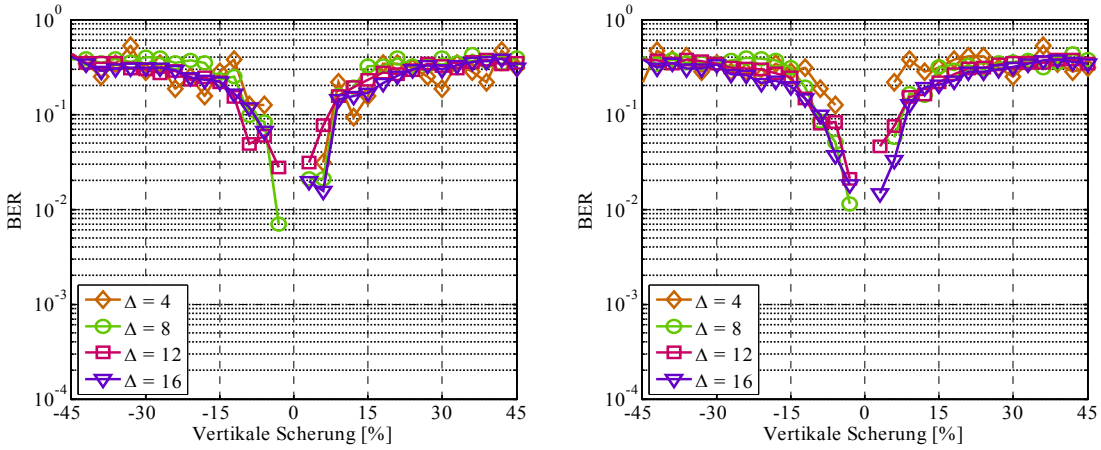


Abbildung H.20 Robustheit gegenüber vertikaler Scherung des Bildes mit den Simulationsparametern: $\sigma_{\min} = 3$, $\sigma_{\max} = 5$ sowie $G_{base} = 5$ (links) und $G_{base} = 10$ (rechts)