Universität des Saarlandes
Naturwissenschaftlich-Technische Fakultät I
Fachrichtung Informatik

Staatsexamensarbeit

Kompression dreidimensionaler Daten mit anisotropen Diffusionsprozessen

eingereicht von

Pascal Tobias Peter

am 13. Januar 2011

Betreuer
Prof. Dr. Joachim Weickert

Prüfer
Prof. Dr. Joachim Weickert
PD Dr. Michael Breuß

Mathematical Image Analysis Group

Peter, Pascal Tobias
Kompression dreidimensionaler Daten mit anisotropen Diffusionsprozessen
Staatsexamensarbeit
Universität des Saarlandes
Saarbrücken
Januar 2011

Kurzzusammenfassung

Durch den steigenden Bedarf an Datenaustausch über das Internet und ständig wachsende Datenmengen gewinnt die effiziente Speicherung dreidimensionaler Bilddaten an Bedeutung. Insbesondere in der Medizin werden täglich große Mengen von 3D-Daten erfasst. Im Verlauf der letzten Jahre haben sich, beginnend mit der Veröffentlichung von Galić et al. (2005), Kompressionsmethoden auf der Basis partieller Differentialgleichungen (PDE, engl. *partial differential equation*) als konkurrenzfähige Alternative zu vorherrschenden Transformationsverfahren etabliert. Für dreidimensionale Bilddaten existieren jedoch noch keine PDE-basierten Alternativen zu den klassischen Kompressionsmethoden. Daher wird in dieser Arbeit der Versuch unternommen, erfolgreiche Konzepte PDE-basierter Kompression auf dreidimensionale Daten zu übertragen.

Um dieses Ziel zu erreichen, wird der zweidimensionale R-EED-Algorithmus von Schmaltz et al. (2010) auf ein dreidimensionales Verfahren erweitert. Diese Erweiterung verwendet unter Einbeziehung der zusätzlichen Dimension ungenutztes Potential zur verlustfreien und verlustbehafteten Kompression, um Qualitätssteigerungen zu erreichen. Das Grundprinzip des R-EED-Algorithmus bleibt dabei erhalten: durch adaptive Bildunterteilung wird zunächst systematisch eine Teilmenge der Bildpunkte des Originals bestimmt und durch klassische Kompressionsmethoden effizient gespeichert. Im Zuge der Bildunterteilung wird diese Menge bekannter Punkte derart gewählt, dass das Ursprungsbild mit Hilfe kantenverstärkender Diffusion (EED, engl. *edge-enhancing diffusion*) möglichst originalgetreu rekonstruiert werden kann. Der Rest der Bildinformationen kann somit verworfen werden.

Für die dreidimensionale Erweiterung wird die adapative Rechtecksunterteilung (engl. *rectangular subdivision*) der R-EED-Methode durch eine adaptive Quaderunterteilung (engl. *cuboidal subdivision*) ersetzt. Dadurch wird zum einen ermöglicht, verlustfreie Kompression durch Entropie-Codierung effizienter zu gestalten und zum anderen die Möglichkeit eröffnet, die Qualität der komprimierten Bilder durch eine Erweiterung des EED-Inpaintings auf drei Dimensionen zu verbessern.

Das Potential des resultierenden C-EED-Verfahrens wird durch Experimente mit synthetischen Testbildern und medizinischen Realdaten belegt. Die Testergebnisse demonstrieren, dass C-EED durch dreidimensionales Inpainting signifikante Qualitätssteigerungen gegenüber R-EED und dem Kompressionsstandard für medizinische Bilddaten erreichen kann.

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Saarbrücken, 11. Januar, 2011

Pascal Tobias Peter

Danksagung

Mein herzlicher Dank gebührt zunächst Prof. Dr. Joachim Weickert für die Vergabe eines spannenden Themas und die Unterstützung durch Rat und Tat. Ohne die unkomplizierte Bereitstellung passender Hardware wären viele der in dieser Arbeit durchgeführten Experimente nicht möglich gewesen. Für die Hilfestellungen bezüglich des Umgangs mit seiner Referenzimplementierung und für sein offenes Ohr für Probleme möchte ich zudem Christian Schmaltz danken.

Bei Dr. Michael Breuß möchte ich mich in seiner Funktion als Zweitprüfer bedanken. Zudem gilt ihm und allen anderen Mitgliedern der Mathematical Image Analysis Group, mit denen ich in den letzten Jahren zusammenarbeiten durfte, mein Dank für ihre Hilfsbereitschaft und die angenehme Arbeitsatmosphäre.

Für ihre gewissenhafte Arbeit als Lektoren danke ich Patrick Trampert und Haiko Wick. Mein besonderer Dank gilt darüber hinaus Sarah Diehl, sowohl für das Redigieren der Arbeit und ihre Hilfe in LaTeX- und Automatisierungs-Fragen, als auch für die moralische Unterstützung.

Zudem möchte ich mich bei meinen Eltern bedanken: zum einen dafür, dass sie mir geholfen haben, diese Arbeit aus dem Blickwinkel eines Informatik-Laien zu betrachten, zum anderen für ihre bedingungslose Unterstützung während meines gesamten Studiums.

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Motivation

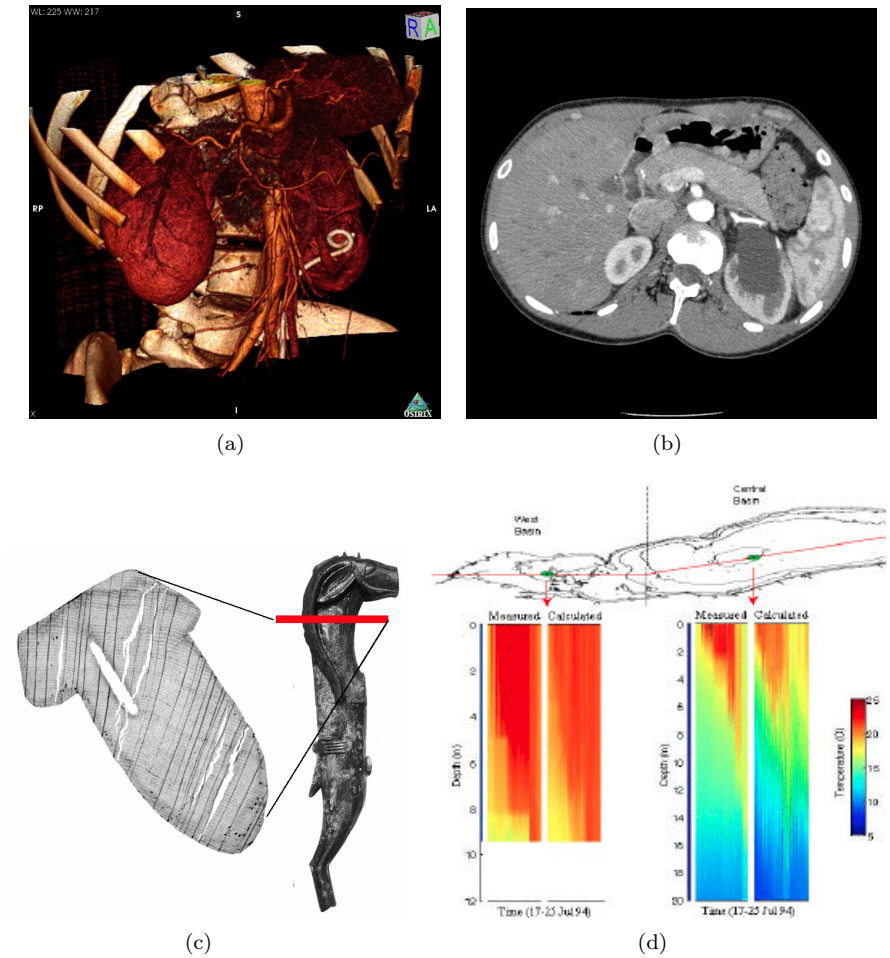
Dreidimensionale Bilddaten sind sowohl in verschiedensten Gebieten der Forschung, als auch im Alltagsleben zu einer festen Komponente der Datenverarbeitung und des Datenaustausches geworden. Aufgrund des technischen Fortschritts wachsen diese Datenmengen ständig, was vor allem im Kontext der Datenübertragung mittels Internet platzeffiziente Speichermethoden erfordert.

Ein durch Computertomographie erstelltes 3D-Bild in einer moderaten Auflösung von $512 \times 512 \times 256$ Bildpunkten hat bei 8 bit Speicherplatzbedarf pro Bildpunkt bereits eine Größe von über 67 Megabyte. Setzt man bei Privatanwendern gängige Übertragungsgeschwindigkeiten von 1-10 Mbit/s voraus, so dauert eine Übertragung dieses Bildes zwischen einer und acht Minuten. Bezieht man umfangreiche Bilddatenbanken oder hochauflösende Videos als dreidimensionale Bilddaten im weiteren Sinne in diese Überlegung ein, so bedeutet die Übertragung der anfallenden Datenmenge auch für schnellere Verbindungen lange Wartezeiten.

Die Lösung dieses Problems besteht in effizienter digitaler Codierung der Daten und der Elimination von Bildinformationen, die für die menschliche Wahrnehmung irrelevant sind. Diese *Kompression* hat zum Ziel, den benötigten Speicherplatz zu reduzieren und dabei den Verlust der Bildqualität gegenüber dem unkomprimierten Original möglichst gering zu halten. Die folgenden Abschnitte illustrieren anhand von Anwendungsbeispielen die Verbreitung dreidimensionaler Bilder in Wissenschaft und Alltag und geben einen Überblick über etablierte Kompressionsstandards.

Praktische Anwendungen

Dreidimensionale Rasterbilder, die sich aus quaderförmigen Volumenelementen (*Voxeln*) mit unterschiedlichen Grau- oder Farbwerten zusammensetzen, sind das weit verbreitete Ausgabeformat vieler bildgebender Verfahren. Insbesondere in der Medizin etablierte sich in den 1970er Jahren durch die Erfindung der *Computertomographie* (CT) die digitale Speicherung und Verarbeitung medizinischer Daten. Um Medizinern möglichst optimalen Zugang zu diagnostischen Daten zu ermöglichen, werden heutzutage neben der auf Röntgenstrahlung



(a) (b)

(c) (d)

Abbildung 1.1: **Dreidimensionale Rasterbilder in der Praxis.** Abb. (a) zeigt eine Volume-Rendering-Visualisierung eines nach dem DICOM-Standard gespeicherten CT-Scans. Ein Schnittbild der zugehörigen Originaldaten ist in Abb. (b) zu sehen (OsiriX, 2009). Abb. (c) enthält ein CT-Schnittbild einer keltischen Figur. Archäologen können so das Innere beschädigungsfrei analysieren (Keefer et al., 2006). Ein Ausschnitt aus tiefenabhängigen Temperaturdaten eines Sees ist in Abb. 1.1(d) dargestellt. Für zwei exemplarische Messpunkte im See wird farbig die Temperatur abhängig von Zeit und Tiefe visualisiert. Dabei werden Messdaten (jeweils links) und Simulationsdaten (rechts) verglichen (León et al., 2005).

basierenden CT auch *Magnet-Resonanz-Tomographie* (MRT) und *Positronen-Emissions-Tomographie* (PET) eingesetzt. Auf diese Art erfasste 3D-Daten werden durch Methoden der Bildverarbeitung aufbereitet und mit Hilfe von Computergrafik visualisiert. Durch Volume-Rendering stellen Visualisierungsprogramme wie die Open Source Software OsiriX (Rosset et al., 2004) 3D-Voxelgrafiken interaktiv dar (siehe Abb. 1.1). Eine Einführung zu bildgebenden Verfahren und Bildverabeitung in der Medizin ist in “Fundamentals of Digital Imaging in

Medicine" von Bourne (2010) zu finden.

Den Einsatz von 3D-Bilddaten jedoch nur im medizinischen Kontext zu betrachten, wäre zu kurz gegriffen. In allen Bereichen der Wissenschaft und Industrie, die bildgebende Verfahren einsetzen oder Visualisierungen diskreter Simulationsdaten nutzen, werden auch dreidimensionale Rasterbilder verwendet. Die vielfältigen Anwendungsmöglichkeiten seien hier durch eine Auswahl von Beispielen belegt: Archäologen verwenden Voxeldaten aus Bodenradar-Messungen (Novo et al., 2008) zur Analyse von Grabungsorten oder können mittels CT-Untersuchung das Innere von Fundstücken beschädigungsfrei untersuchen (Keefer et al., 2006). Als Beispiele für die dreidimensionale Visualisierung von Simulationsdaten seien thermodynamische Modelle in der Meteorologie (León et al., 2005) und Partikelinteraktionen in der Physik (Ketcham und Feder, 2003) genannt.

Während sich die bisherigen Beispiele auf den praktischen Einsatz von 3D-Bilddaten durch Forscher und Experten beziehen, lässt sich auch eine Verbindung zum Alltag breiter Bevölkerungsteile herstellen: Bildsequenzen lassen sich auch als Voxeldaten mit zwei räumlichen und einer zeitlichen Koordinatenachse interpretieren. Insbesondere im Kontext der wachsenden Verbreitung hochauflösender Videos über das Internet ist eine starke Kompression der Inhalte im Vergleich zur Distribution via Datenträger (z.B. DVD, BluRay) von großer Bedeutung. Hierbei ist jedoch zu beachten, dass Videodaten aufgrund der Unterscheidung von Raum- und Zeitdimension andere Eigenschaften besitzen als die Bilddaten mit drei gleichberechtigten Dimensionen.

Zusammenfassend lässt sich anhand der genannten Beispiele schließen, dass die Suche nach neuen, effektiven Kompressionsmethoden für dreidimensionale Bilddaten aktuell für viele Anwendungsgebiete relevant ist und in Zukunft voraussichtlich noch an Bedeutung gewinnt.

Klassische Kompressionsmethoden

Alle Verfahren der Bildkompression lassen sich in die Kategorien *verlustfreie* und *verlustbehaftete* Kompression einordnen. Verlustfreie Kompression speichert die im Bild enthaltenen Informationen lediglich auf effizientere Weise, während verlustbehaftete Kompression gezielt Informationen verwirft, die für die menschliche Wahrnehmung des Bildes irrelevant oder weniger wichtig sind. Auf diese Weise lässt sich eine bessere *Kompressionsrate* erzielen, d.h. der Speicherplatzbedarf wird weiter verringert als bei verlustfreien Verfahren. Im Fokus dieser Arbeit stehen verlustbehaftete Verfahren, deren Resultate auch für hohe Kompressionsraten angemessene Bildqualität aufweisen.

Bildkompression bei bildgebenden Verfahren in der Medizin folgt in der Regel dem DICOM Standard (National Electrical Manufacturers Association, 2004), welcher neben verlustfreien Kompressionsmethoden auch verlustbehaftete Methoden wie JPEG (Pennebaker und Mitchell, 1992) und JPEG2000 (Taubman und Marcellin, 2002) zulässt. Für Videokompression werden zur Zeit im praktischen Gebrauch in erster Linie die Kompressionsverfahren MPEG-4 Part 2 (Moving Picture Experts Group, 2004) und H.264/MPEG-4 AVC (Moving Picture Experts Group, 2009) eingesetzt.

Die genannten verlustbehafteten Kompressionsverfahren gehören der Klasse der *Transformationscodierer* an. Ziel dieser Methoden ist es, die Bilddarstellung durch Grau- oder Farbwerte mit Hilfe einer Transformation in eine neue Reprä-

Abbildung 1.2: **Frequenztransformation.** Die Abbildung zeigt die Zerlegung eines Originalbildes (rechts) in zwei Basisbilder (links und Mitte). Die Summe der beiden Kosinus-Wellen der Basisbilder ergibt das Rautenmuster des Originalbildes. Mit einer entsprechenden Vielzahl von Basisbildern kann auch ein komplexes Bild (z.B. ein Foto) durch eine Frequenztransformation repräsentiert werden.

sentation zu überführen, die die Bildinformationen verdichtet darstellt. Dieses abstrakte Konzept lässt sich anhand konkreter Anwendungen nachvollziehen: Im Falle der in JPEG und MPEG-4 verwendeten *diskreten Kosinus-Transformation* (DCT, engl.: *discrete cosine transform*) wird das Bild in sich überlagernde Wellenfunktionen mit unterschiedlicher Frequenz zerlegt. Statt für jeden Bildpunkt einen Grauwert zu speichern, wird das Bild als gewichtete Summe von Basisbildern interpretiert (siehe Abb. 1.2). An die Stelle der örtlich lokalisierten Grauwerte treten die globalen Gewichte der Basisbilder. Diese Koeffizienten erlauben es, für die visuelle Wahrnehmung weniger wichtige Bildinformationen zu identifizieren. Die niedrigen Frequenzen einer DCT stellen grobe Strukturen im Bild dar, während die hochfrequenten Anteile in erster Linie kleine Details und Bildrauschen enthalten.

Durch die gezielte Entfernung weniger wichtiger Informationen aus den Koeffizienten der Basisbilder wird eine verlustbehaftete, effiziente Speicherung realisiert. Konkret zerlegt JPEG zweidimensionale Eingabebilder in Blöcke der Dimension 8×8 und führt auf jedem dieser Blöcke eine *Quantisierung* der DCT-Koeffizienten durch. Dabei werden die kontinuierlichen Koeffizienten durch diskrete Werte ersetzt. Gewichte der hoch- und niederfrequenten Basisbilder werden dabei unterschiedlich behandelt und können durch sogenannte *Entropie-Codierung* (siehe Kapitel 2.4) effizient gespeichert werden. Die räumliche Unterteilung in kleine Teilbilder ist notwendig, da im Allgemeinen verschiedene Bildregionen sehr unterschiedliche Verteilungen der DCT-Koeffizienten aufweisen, die Koeffizienten selbst jedoch für das gesamte Bild gültig sind. Die Blockunterteilung sorgt somit für eine künstliche Verortung der Gewichte. Gleichzeitig ergibt sich dadurch jedoch ein gravierender Nachteil: bei hoher Kompressionsrate entstehen Diskontinuitäten an den Rändern der Teilbilder, die sich in deutlich sichtbaren Blockartefakten äußern (vgl. Abb. 1.3(d)).

Im Wesentlichen kommt das bisher beschriebene Transformationsverfahren auch in beiden MPEG-4-Verfahren zur Anwendung, wobei bei H.264 eine andere, von der DCT abgeleitete Transformation auf kleineren Blöcken (4×4) eingesetzt wird. JPEG2000 hingegen unterscheidet sich durch die Wahl der *diskreten Wavelet-Transformation* (DWT) deutlich von den vorgenannten Verfahren. *Wavelet* lässt sich frei als "kleine Welle" übersetzen, ein Name, der die Verwandtschaft zu Wellenfunktionen nahelegt. Die Basisfunktionen der DWT haben je-

doch mehrere Vorteile gegenüber denen der DCT: Zum einen sind die Funktionen der DWT im Gegensatz zu periodischen Kosinusfunktionen beschränkt, zum anderen sind sie so gewählt, dass die einzelnen Koeffizienten auch Ortsbezüge enthalten und eine künstliche räumliche Beschränkung nicht notwendig ist. Obwohl JPEG2000 im Kern ebenfalls auf einer geschickten Quantisierung der Transformationskoeffizienten beruht, werden Blockartefakte durch die DWT vermieden und höhere Kompressionsraten bei guter Qualität erzielt.

Obwohl der Transformationsschritt im Zusammenspiel mit Quantisierung und Entropie-Codierung das Kernstück der klassischen verlustbehafteten Kompressionsalgorithmen darstellt, enthalten diese Verfahren eine Vielzahl zusätzlicher verlustfreier und verlustbehafteter Kompressionsschritte wie *Prädikation* oder die Ausnutzung von *Farbraumeigenschaften*. MPEG-4 und H.264 verfügen darüber hinaus über ein breites Arsenal an auf bestimmte Bildinhalte spezialisierte Methoden wie Bewegungskompensation. Für einen detaillierten Überblick über die genannten klassischen Kompressionsverfahren sei auf das Buch von Strutz (2002) verwiesen.

PDE-basierte Kompression

Transformationsbasierte Kompressionsmethoden wie JPEG und MPEG sind zwar weit verbreitet und erfolgreich, stellen jedoch nicht die einzigen bewährten Ansätze für Bildkompression dar. Galić et al. (2005, 2008) entwickelten ein Kompressionsverfahren auf der Basis von Interpolation unter Verwendung partieller Differentialgleichungen (PDE, engl. *partial differential equation*). Die Kernidee dieser Methode besteht darin, systematisch einen geringen Anteil der Punkte des Ausgangsbildes auszuwählen und das Originalbild aus diesen bekannten Punkten zu rekonstruieren. Ausgehend von diesem Grundansatz wurden im Laufe der letzten Jahre konkurrenzfähige Alternativen zu den vorgenannten Algorithmen entwickelt.

Schmaltz et al. (2009, 2010) gelang es mit Hilfe einer adaptiven Baumstruktur einen Bruchteil der Bildpunkte eines zweidimensionalen Ausgangsbildes derart auszuwählen, dass mit Hilfe von Interpolation JPEG2000 bei gleicher Kompressionsrate qualitativ übertroffen wurde. Die besten Interpolationsergebnisse wurden hierbei mit Hilfe PDE-basierter Interpolation erzielt. Insbesondere ein aus physikalischen Vorgängen abgeleitetes PDE-Modell *kantenverstärkender Diffusion* (EED, engl. *edge-enhancing diffusion*) erwies sich als gut geeignet für die Bildrekonstruktion. Eine Einführung in die Funktionsweise des sogenannten R-EED-Verfahrens folgt in Abschnitt 1.2 und detaillierte Erläuterungen zum theoretischen Hintergrund der Teilschritte des Algorithmus sind in Kapitel 2 zu finden.

Ziele

Da sich R-EED bereits im zweidimensionalen Fall als konkurrenzfähig zu JPEG-2000 erwiesen hat und JPEG2000 als Teil des DICOM-Standards einen Maßstab für die Kompression dreidimensionaler Rasterbilder darstellt, liegt die Anwendung des R-EED-Verfahrens auf 3D-Voxeldaten nahe.

Ziel dieser Arbeit ist es, das Potential PDE-basierter Diffusion im Sinne von Schmaltz et al. (2009) für die Kompression dreidimensionaler Bilddaten auszuloten. Dazu wird das zweidimensionale R-EED-Verfahren um dreidimensionale,

(a) (b)

(c) (d)

Abbildung 1.3: **PDE-basierte Kompression** Abb. (c) zeigt das Ergebnis einer Kompression des Originalbildes (a) durch das R-EED Verfahren im Verhältnis 60:1. Dieses Bild wurde aus den bekannten Punkten der Inpaintingmaske (b) mittels diffusionsbasierter Interpolation rekonstruiert. In Abb. (d) ist ein JPEG-komprimiertes Bild mit gleicher Kompressionsrate zu sehen (Quelle: Schmaltz et al. (2010)).

adaptive Bildunterteilung und diffusionsbasierte 3D-Interpolation erweitert.

In besonderem Maße soll untersucht werden, ob und wie diese Erweiterung Redundanzen in der dritten Dimension ausnutzen kann, um bei gleichem Kompressionsverhältnis eine höhere Bildqualität als eine sequentielle Schnittbildkompression mit R-EED zu erreichen. Zusätzlich werden auch Vergleiche zur Qualität von JPEG2000 gezogen, im Vordergrund steht jedoch die Gegenüberstellung mit R-EED.

1.2 Problemstellung und C-EED-Ansatz

Zweidimensionale Situation

Wie bereits in Abschnitt 1.1 angedeutet, beruht das R-EED-Kompressionsverfahren von Schmaltz et al. (2009, 2010) auf PDE-basierter Interpolation. Nur wenige Punkte eines Bildes werden gespeichert, aus denen das Ursprungs-

bild möglichst originalgetreu rekonstruiert wird. Diese Rekonstruktion wird als *Inpainting* bezeichnet. Die Aufteilung des Originalbildes in bekannte und zu rekonstruierende Punkte wird durch die zugehörige *Inpaintingmaske* festgelegt (siehe Abb. 1.3(b)).

Grundsätzlich gilt es, gleichzeitig die Qualität der Rekonstruktion und die Menge der gespeicherten Punkte zu optimieren. Die Rekonstruktionsqualität hängt zunächst von der Wahl der Inpaintingmethode ab. Die Ergebnisse von Schmaltz et al. (2010) legen nahe, dass PDE-basiertes Inpainting im Allgemeinen und kantenverstärkende Diffusion im Besonderen gut für diese Aufgabe geeignet sind.

Um optimale Rekonstruktionen aus einer bestimmten Anzahl von Teilpunkten zu erhalten, müssen nicht nur die Inpainting-Methode und deren Parameter passend gewählt werden, sondern auch die Inpaintingmaske. Zwar liegen für bestimmte Inpaintingmethoden theoretische Ergebnisse zu den Zusammenhängen zwischen der Qualität von Inpaintingergebnissen und der Wahl der bekannten Punkte vor (Belhachmi et al., 2009), im Kontext der Bildkompression ist allerdings neben qualitativen Aspekten auch der Speicherplatzbedarf von großer Bedeutung. Zum einen muss neben den Grauwerten der bekannten Punkte auch deren Position im Bild gespeichert werden, zum anderen kommt es zu Wechselwirkungen zwischen der Wahl der Punkte und der effizienten Codierung der zugehörigen Grauwerte. Daher ist eine optimale Auswahl der Punkte in Bezug auf den Inpaintingschritt nicht zwangsläufig eine optimale Wahl im Sinne des gesamten Kompressionsverfahrens.

Um diesen Umständen Rechnung zu tragen, schränken Schmaltz et al. (2010) die Auswahl der bekannten Punkte ein. Anhand eines vorgegebenen Musters werden einige wenige, isolierte Punkte des Bildes gespeichert (z.B. die Eckpunkte des rechteckigen Bildbereichs). Genügt die Rekonstruktion aus diesen Punkten einer a priori festgelegten Fehlerschranke, ist die Interpolationsmaske auf diese Weise gegeben. Ist der Fehler hingegen zu groß, wird das Bild in zwei rechteckige Teilbilder aufgespalten und die Inpaintingmaske durch die Punktmuster der Teilbilder definiert. Diese Bildunterteilung wird rekursiv durchgeführt, bis entweder eine weitere Bildunterteilung nicht möglich ist oder der Rekonstruktionsfehler den Vorgaben entspricht. Die Fehlerschranke wird hierbei der Rekursionstiefe angepasst.

Die räumliche Position der gespeicherten Punkte ist somit durch die Unterteilung des Bildes gegeben und kann in Form eines binären Baumes effizient gespeichert werden. Durch die Regulierung der Baumtiefe mit Hilfe der adaptiven Fehlerschranke wird die Menge und Position der gespeicherten Punkte an lokale Gegebenheiten des Bildes angepasst: in der Nähe semantisch wichtiger Bildelemente wie Kanten werden mehr Punkte gespeichert als in anderen Regionen. Die *adaptive Rechtecksunterteilung* stellt in Kombination mit EED-Inpainting die Kernfunktionalität der R-EED Methode dar, wenngleich der Algorithmus durch zusätzliche Schritte ergänzt wird, die Kompressionsrate und Rekonstruktionsqualität verbessern (siehe Kapitel 2).

Dreidimensionale Erweiterung

Im dreidimensionalen Fall ändert sich an der Problemstellung für EED-Bildkompression nichts Grundlegendes. Es ist ohne weiteres möglich, ein 3D-Bild in Schnittbilder zu zerlegen und durch getrennte Anwendung von R-EED auf

diese Teilbilder zu komprimieren. Dabei wird jedoch nicht der volle Informationsgehalt der Bilder ausgenutzt, um eine möglichst gute Kompressionsrate zu erreichen. Die Schnittbilder sind nicht voneinander unabhängig, sondern stehen in der Regel in engem Zusammenhang zu ihren Nachbarbildern. Zweidimensionale Inpainting-Methoden beruhen auf der Annahme, dass zwischen Bilddaten ein stetiger Zusammenhang besteht und nutzen diesen aus, um unbekannte Bildpunkte aus den bekannten zu rekonstruieren. Wendet man nun R-EED auf Schnittbilder an, werden nur die Zusammenhänge in zwei der drei Richtungen ausgenutzt.

Die zentrale Aufgabe für eine Erweiterung der R-EED-Methode auf drei Dimensionen besteht also in der Nutzung dreidimensionaler Diffusion zur Bildrekonstruktion und der Auswahl einer passenden Inpaintingmaske. Als natürliche Erweiterung der Rechtecksunterteilung von R-EED wird hierbei auf eine adaptive Quaderunterteilung zurückgegriffen. Analog zum zweidimensionalen Fall wird das Originalbild anhand einer an die Rekursionstiefe angepassten Fehlerschranke rekursiv unterteilt und die Inpaintingmaske mit Hilfe dreidimensionaler Punktmuster generiert.

In Anlehnung an das R-EED-Kompressionsverfahren, welches seinen Namen aus den definierenden Elementen der rechteckigen Bildunterteilung (engl. *rectangular subdivision*) und der verwendeten Methode der Bildinterpolation (engl. *edge-enhancing diffusion*) ableitet, wird das neue, dreidimensionale Kompressionsverfahren mit quaderförmiger Bildunterteilung (engl. *cuboidal subdivision*) im Folgenden als C-EED bezeichnet.

Neben der bereits genannten, zentralen Erweiterung des Inpaintingschritts um eine dritte Dimension sind auch die übrigen Kompressionsschritte des R-EED Verfahrens anzupassen und, soweit möglich, weitere durch die Dreidimensionalität gegebene Redundanzen auszunutzen. Eine detaillierte Beschreibung dieser Anpassungen erfolgt in Kapitel 3.

1.3 Gliederung

In den verbleibenden Abschnitten des einleitenden Kapitels wird ein Überblick über verwandte Arbeiten gegeben. Weiterhin werden einige grundlegende mathematische Definitionen und Notationskonventionen festgelegt.

Kapitel 2 enthält eine detaillierte Beschreibung der Funktionsweise und theoretischen Hintergründe des zweidimensionalen R-EED-Algorithmus von Schmaltz et al. (2009, 2010), welcher die Grundlage für das in dieser Arbeit vorgestellte Kompressionsverfahren darstellt. Die behandelten Teilschritte umfassen adaptive Rechtecksunterteilung, PDE-basierte Interpolation, Quantisierung, Entropie-Codierung sowie Grauwertoptimierung und Maskeninversion.

Die Erweiterung des R-EED-Verfahrens auf drei Dimensionen wird in Kapitel 3 beschrieben. Dabei werden die vom neuen C-EED-Algorithmus ausgenutzten Redundanzen erläutert und die Unterschiede zum R-EED-Modell herausgestellt. Schwerpunkte sind hierbei dreidimensionale kantenverstärkende Diffusion sowie adaptive Quaderunterteilung. Zusätzlich wird die Eignung von R- und C-EED für ROI-Codierung (engl. *region of interest*) behandelt. Das darauf folgende Kapitel 4 widmet sich der numerischen Umsetzung des C-EED-Verfahrens. Neben der Diskretisierung der EED-basierten Interpolation werden auch numerische Lösungsverfahren und die konkrete ANSI-C-Implementierung

thematisiert.

Kapitel 5 enthält die Beschreibung von Experimenten auf synthetischen und medizinischen Testbildern sowie die daraus resultierenden Ergebnisse und deren Analyse. Dabei werden die Algorithmen R-EED, C-EED, C-EED2D und JPEG2000 qualitativ verglichen. Die Arbeit schließt in Kapitel 6 mit einer Zusammenfassung der gewonnenen Erkenntnisse und einem Ausblick auf mögliche weiterführende Forschungsansätze.

Ergänzende Bilder zu den Experimenten aus Kapitel 5 sind in Anhang A enthalten. Anhang B hingegen enthält eine Kurzanleitung für die Programme der Referenzimplementierung.

1.4 Verwandte Arbeiten

In den folgenden Abschnitten wird diese Arbeit in die Forschungsfelder PDE-basiertes Inpainting und Bildkompression eingeordnet. Darüber hinaus werden ergänzend einige Transformationscodierungsverfahren genannt, die auf dreidimensionale Bildkompression spezialisiert sind und in Abschnitt 1.1 noch nicht behandelt wurden.

PDE-basiertes Inpainting

Ein wichtiger Grundbaustein der R-EED- und C-EED-Verfahren ist PDE-basiertes Inpainting. Im Wesentlichen beruhen alle PDE-basierten Inpaintingmethoden auf Transportgleichungen, die die Überführung von Grauwertinformationen von bekannten Punkten in unbekannte Bildteile modellieren. Theoretische Hintergründe der durch derartige Transportgleichungen definierten Diffusionsprozesse werden in den Veröffentlichungen von Weickert (1996b, 1999) behandelt.

Die Eignung verschiedener partieller Differentialgleichungen wurde durch eine Vielzahl wissenschaftlicher Arbeiten auf verschiedene Weise belegt. Ausgehend von der axiomatischen Untersuchung der Anwendung partieller Differentialgleichungen zu Interpolationszwecken (Caselles et al., 1998), wurden erfolgreich iterative (Bertalmío et al., 2000, 2001; Battiato et al., 2003; Tschumperlé und Deriche, 2003; Tschumperlé, 2006) und nichtiterative (Telea, 2004; Bornemann und März, 2007) Inpaitingalgorithmen entwickelt. Eine eigene Klasse von Ansätzen bilden dabei Variationsmethoden, die Inpainting als Minimierung von Energiefunktionalen modellieren (Masnou und Morel, 1998; Masnou, 2002; Chan und Shen, 2002a; Esedoglu und Shen, 2002; Roussos und Maragos, 2007). Ein Überblick über verschiedene mathematische Methoden des Image Inpainting ist in "Mathematical Models for Local Nontexture Inpaintings"(Chan und Shen, 2002b) sowie in "Variational Image Inpainting"(Chan und Shen, 2005) zu finden.

Im Kontext der PDE-basierten Bildkompressionsmethoden sind zudem die theoretischen Untersuchungen von Belhachmi et al. (2009) zu nennen, die den engen Zusammenhang zwischen der Auswahl von Ausgangsdaten und Interpolationsresultaten homogener Diffusion behandeln.

Verwandte Kompressionsmethoden

Grundsätzlich lässt sich der Einsatz von Diffusion in kompressionsbezogenen Forschungsarbeiten in zwei Klassen einordnen: diffusionsbasierte Methoden nutzen die Diffusion als zentrales Element der Kompression, während diffusionsunterstützte Verfahren lediglich davon Gebrauch machen, um die Ergebnisse anderer Kompressionsschritte zu verbessern.

Diffusionsbasierte Kompression

Die bereits genannten Kompressionsalgorithmen von Galić et al. (2005, 2008) und Schmaltz et al. (2009, 2010) sind am engsten mit der C-EED-Methode verwandt. R-EED stellt eine Verfeinerung des Verfahrens von Galić et al. (2008) dar. Der augenfälligste Unterschied zwischen den beiden Verfahren besteht in der Form der adaptiven Bildunterteilung. Galić et al. (2008) nehmen eine Dreiecksunterteilung im Sinne von Distasi et al. (1997) vor, während Schmaltz et al. (2010) eine Rechtecksunterteilung vornehmen, die Verfahren mit adaptiver Blockgröße wie den Methoden von Strobach (1991) und Sullivan und Baker (1994) ähnelt. Weitere Unterschiede liegen in zusätzlichen Kompressionsschritten sowie einer verbesserten Parameteranpassung im Inpainting-Schritt des R-EED-Verfahrens.

Ein weiterer diffusionsbasierter Ansatz stammt von Mainberger und Weickert (2009). Dieses Verfahren ist auf cartoonartige Bilder spezialisiert und kann in diesem Anwendungsbereich JPEG2000 übertreffen. Die Auswahl der Inpaintingmaske wird bei Mainberger und Weickert (2009) durch Kantendetektion getroffen. Da Cartoonbilder bis auf die Bildkanten zum Großteil aus einheitlichen Flächen bestehen, reicht die Speicherung von Bildpunkten nahe der Kanten aus, um gute Inpainting-Ergebnisse mit homogener Diffusion zu erzielen. Dieser Ansatz hat den Vorteil, dass homogene Diffusion weniger Rechenzeit erfordert als EED und zudem die Implementierung erleichtert wird.

In Hinblick auf den Einsatz PDE-basierter Kompressionsmethoden auf dreidimensionale Daten im weitesten Sinne existieren bisher nur Anwendungen die auf Einzelbildkompression von Videosequenzen beruhen. Köstler et al. (2007) entwickelten auf der Basis von Mehrgitterverfahren und dem Verfahren von Galić et al. (2005) einen auf einer Playstation 3 in Echtzeit einsetzbaren Videokompressionsalgorithmus.

Diffusionsunterstützte Kompression

Wenngleich Diffusionsprozesse auf 3D-Bilddaten bisher nicht direkt zur Kompression eingesetzt werden, finden sie dennoch Anwendung im Bereich der Bildverbesserung. Grundsätzlich gibt es zwei Ansatzpunkte für die Verbesserung von Kompressionsergebnissen transformationsbasierter Methoden durch Diffusion: Zum einen erzeugen JPEG und MPEG inhärent ungewollte Bildstörungen, insbesondere Blockartefakte. Mit Diffusion können diese Artefakte nachträglich entfernt oder ihre optische Wirkung gemindert werden (Ford, 1996; Yang und Hu, 1997; Gothandaraman et al., 2001; Alter et al., 2005; Tsuji et al., 2002).

Auf der anderen Seite kann Diffusion schon vor dem Kompressionsschritt unerwünschte Bildelemente wie Bildrauschen herausfiltern oder die Koeffizienten der Basisfunktionen vorteilhaft beeinflussen (Ford et al., 1992; Ford, 1996; Chan

und Zhou, 1998; Bourdon et al., 2004; Kopilovic und Szirányi, 2005; Tsuji et al., 2007).

Die meisten diffusionsunterstützten Kompressionsansätze beruhen auf 2D-Diffusion, einige der genannten Methoden setzen jedoch auch dreidimensionale Diffusion (Tsuji et al., 2002; Bourdon et al., 2004) ein.

Transformationsbasierte 3D-Kompression

Transformationsbasierte Bildkompression beschränkt sich nicht auf rein zweidimensional agierende Verfahren wie JPEG und JPEG2000. Bereits seit der Veröffentlichung von Karlson und Vetterli (1988) werden auch Ansätze erforscht, die auf Transformationscodierung mit separablen 3D-Wavelets beruhen (Baskurt et al., 1995; Kim und Pearlman, 1997, 1999; Secker und Taubman, 2002; Lewis und Knowles, 1990). Diese Methoden erweitern zweidimensionale Transformationsverfahren in ähnlicher Weise wie R-EED durch C-EED erweitert wird. Da es jedoch eine Vielzahl solcher Methoden gibt und diese keine Standardisierung wie im DICOM-Standard aufweisen, wird in dieser Arbeit auf vergleichende Experimente verzichtet.

Darüber hinaus wird im Bereich der medizinischen Bildverarbeitung zudem vor allem spezialisierte Forschung mit praktischen Bezügen betrieben. Eine wichtige Funktionalität von Bildkompressionsmethoden in der Medizin ist die Speicherung einzelner Bildregionen in unterschiedlicher Qualität (Lehmann et al., 1999; Gokturk et al., 2001). Diese sogenannte *ROI-Codierung* (engl. *region of interest*) wird eingesetzt, um hohe Bildqualität in diagnoserelevanten Regionen medizinischer Bilder zu garantieren und gleichzeitig die Kompressionsrate auf Kosten weniger wichtiger Bildteile zu reduzieren. Ein weiteres Beispiel für anwendungsbezogene Spezialisierung von Kompressionsverfahren ist die Erhöhung der Zugriffsgeschwindigkeit auf beliebige Bilddaten für Interaktive Anwendungen, die auf großen Datensätzen operieren (Bajaj et al., 2001).

1.5 Definitionen und Notationskonventionen

Variablen und Operatoren

Weit verbreitenden Notationskonventionen folgend werden in dieser Arbeit skalare Werte, Funktionen und Vektoren durch kleine Buchstaben gekennzeichnet, Matrizen und Mengen durch Großbuchstaben. Um die Unterscheidung zwischen vektorwertigen und skalaren Variablen zu erleichtern, werden Vektoren durch Fettschrift hervorgehoben:

$$x \in \mathbb{R}, \qquad \mathbf{v} \in \mathbb{R}^n$$

Mathematische Operationen wie Addition oder Multiplikation zwischen Funktionen werden wie üblich punktweise definiert. Für einen binären Operator $\diamond \in \{+, -, \cdot, /\}$ gilt

$$(f \diamond g)(x) = f(x) \diamond g(x)$$

Skalarprodukte bezeichnen in dieser Arbeit immer das euklidische Skalar-

produkt und Vektornormen die dadurch induzierte euklidische Norm:

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n : \quad < \mathbf{x}, \mathbf{y} > := \mathbf{x}^T \mathbf{y} = \sum_{i=1}^{n} x_i y_i$$

$$\|\mathbf{x}\| := \sqrt{\mathbf{x}^T \mathbf{x}} = \sqrt{\sum_{i=1}^{n} x_i^2}$$

Aufgrund der Notationskonventionen für Variablen entfällt in vielen Fällen die Notwendigkeit expliziter Kennzeichnung von Operatoren durch unterschiedliche Symbole. So lässt sich beispielsweise anhand der Notation der Variablen eindeutig ersehen, ob der Multiplikationsoperator $\cdot$ eine Skalarmultiplikation (z.B. αx), ein Skalarprodukt (z.B. $\mathbf{x}^T \mathbf{y}$) oder eine Matrix-Vektormultiplikation (z.B. $M\mathbf{x}$) darstellt.

Weiterhin ist die Abrundungsfunktion, auch als Gaußklammer bezeichnet, definiert als:

$$\lfloor x \rfloor := \max_{k \in \mathbb{Z}, k \leq x} k$$

Abschließend ist die Definition des *Faltungsoperators* zu nennen, der in dieser Arbeit im Kontext von Gauß-Glättung und linearer Diffusion verwendet wird. Die Faltung zweier Funktionen $f, g : \mathbb{R}^n \to \mathbb{R}$ ist definiert als:

$$(f * g)(x) := \int_{\mathbb{R}^n} f(y) g(x - y) dy$$

Ableitungen und Differentialoperatoren

Da partielle Differentialgleichungen einen wesentlichen Anteil des theoretischen Grundgerüsts des C-EED-Verfahrens ausmachen, werden in dieser Arbeit *partielle Ableitungen* häufig verwendet. Je nach Kontext wird eine Reihe verschiedener Notationen eingesetzt, um die Lesbarkeit zu verbessern. Für $f : \mathbb{R}^n \to \mathbb{R}$ sind folgende Schreibweisen äquivalent:

$$\frac{\partial}{\partial x_i} f(\mathbf{a}) = \partial x_i f(\mathbf{a}) = f_{x_i}(\mathbf{a}) := \lim_{h \to 0} \frac{f(a_1, \ldots, a_i + h, \ldots, a_n) - f(\mathbf{a})}{h}$$

Die in der Arbeit betrachteten Funktionen sind in der Regel stetig differenzierbar. Die Zugehörigkeit einer Funktion f zur Klasse n-mal stetig differenzierbarer Funktionen wird dargestellt durch $f \in C^n$.

Neben partiellen Ableitungen sind der *Gradient* ∇f sowie die *Divergenz* div$\mathbf{f}$ häufig verwendete Differentialoperatoren. Sie sind definiert als:

$$\nabla f(\mathbf{a}) := \left(\frac{\partial}{\partial x_1} f(\mathbf{a}), \ldots, \frac{\partial}{\partial x_n} f(\mathbf{a}) \right)^T, \quad f : \mathbb{R}^n \to \mathbb{R}$$

$$\text{div} f_i(\mathbf{a}) := \sum_{i=1}^{n} \frac{\partial}{\partial x_i} \mathbf{f}(\mathbf{a}), \quad \mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$$

Charakteristische Funktion und Rand

In dieser Arbeit wird die charakteristische Funktion als Indikatorfunktion für Mengenzugehörigkeiten verwendet. Für eine beliebige Menge A ist die charakteristische Funktion $\chi_A : A \to \{0, 1\}$ von A definiert:

$$\chi_A(x) := \begin{cases} 1, & x \in A \\ 0, & \text{andernfalls} \end{cases}$$

Funktionale Interpretation von Bilddaten

Dreidimensionale Bilder werden in dieser Arbeit als Funktion interpretiert, die jedem Bildpunkt (x, y, z) der Bildpunktmenge Ω einen Grauwert aus der Grauwertmenge G zuordnen:

$$u : \Omega \subset \mathbb{R}^3 \to G \subset \mathbb{R}, \quad (x, y, z) \mapsto u(x, y, z)$$

Die Bildpunktmenge Ω hat für die in dieser Arbeit betrachteten Bilder die Form eines Quaders der Dimension $d_x \times d_y \times d_z$. Daher lässt sich die Bildpunktmenge als kartesisches Produkt dreier Intervalle darstellen:

$$\Omega = [a_x; b_x] \times [a_y; b_y] \times [a_z; b_z] \text{ mit } b_i - a_i = d_i, \quad i \in \{x, y, z\}$$

Im Zuge der Diskussion theoretischer Hintergründe werden für die betrachteten Bilder beliebig häufige stetige Differenzierbarkeit angenommen ($u \in C^\infty$).

In der Praxis sind sowohl die Eingabe- als auch die Ausgabedaten des Kompressionsverfahrens jedoch keine *kontinuierlichen* Funktionen, sondern *diskret*: Grauwerte sind nur für die Pixel des Eingabebildes, also für eine endliche Teilmenge von Ω bekannt. Das diskrete Bild geht aus der kontinuierlichen Funktion u durch sogenannte *Abtastung* hervor. Dabei wird für jedes der drei Intervalle $I_j = [a_j; b_j]$, $j \in \{x, y, z\}$, die Ω definieren, eine endliche Auswahl von Punkten $a_j = p_1^j < p_2 < \cdots < p_{n_j}^j = b_j$ betrachtet. Aufeinanderfolgende Punkte haben dabei alle den gleichen Abstand $h_j := p_i^j - p_{i-1}^j = \frac{d_j}{n_j}$.

Die Punkte $(p_i, p_j, p_k)^T \in \{p_1^x, \ldots, p_{n_x}^x\} \times \{p_1^y, \ldots, p_{n_y}^y\} \times \{p_1^z, \ldots, p_{n_z}^z\}$ definieren ein dreidimensionales, *regelmäßiges Gitter* im Raum. Hierbei wird h_j als *Gitterweite* in j-Richtung bezeichnet. Das diskrete Bild ergibt sich nun aus den Grauwerten der Bildfunktion u an den Gitterpunkten:

$$\forall (i, j, k)^T \in \{1, \ldots, n_x\} \times \{1, \ldots, n_y\} \times \{1, \ldots, n_z\} : \quad u_{i,j,k} := u(p_i, p_j, p_k)$$

$u_{i,j,k}$ bezeichnet also den Grauwert des Pixels mit Index i, j, k und die Anzahl der Gitterpunkte in jeder Richtung bestimmt die *Auflösung* $n_x \times n_y \times n_z$ des Bildes.

Kapitel 2

Das R-EED-Verfahren

In diesem Kapitel wird der R-EED-Algorithmus von Schmaltz et al. (2009, 2010) für zweidimensionale Bilder beschrieben. R-EED bildet die Grundlage für die neue, in Kapitel 3 und 4 behandelte C-EED-Methode.

Das grundlegende Prinzip des R-EED-Verfahrens beruht darauf, lediglich eine sorgfältig ausgewählte Teilmenge $B \subset \Omega$ der Bildpunkte des Originalbildes zu speichern und aus dieser durch Inpainting eine Näherung des Originals zu rekonstruieren. Dieses Kernstück des Algorithmus wird durch klassische Kompressionsmethoden und qualitätssteigernde Zusatzschritte erweitert. In den folgenden Abschnitten 2.1 bis 2.6 werden zunächst die Grundbausteine des Algorithmus erläutert. Deren Zusammenwirken ist Gegenstand von Abschnitt 2.7.

2.1 Inpainting

Ausgangspunkt für das R-EED Verfahren von Schmaltz et al. (2009) und verwandte Kompressionsmethoden (Galić et al., 2008; Mainberger und Weickert, 2009) ist PDE-basiertes *Inpainting*, eine spezielle Form der Interpolation. Inpaintingalgorithmen stellen aus einer Menge $B \subset \Omega$ bekannter Bildpunkte eine Rekonstruktion des Originalbildes her. B wird *Bildbereich* oder *Inpaintingmaske* genannt, während die unbekannte Bildregion $\Omega \setminus B$ als *Inpaintingbereich* bezeichnet wird.

In der Regel wird Inpainting zum Zweck der Bildrestauration eingesetzt, d.h. um beschädigte oder fehlende Bildteile wiederherzustellen (z.B. Kratzer auf Fotos). Während in diesen Fällen die Menge der bekannten Punkte große Teile des Originalbildes umfasst, werden im Kompressionskontext nur sehr wenige Punkte gespeichert (z.B. weniger als 20%).

Weickert und Welk (2006) zufolge lässt sich das Interpolationsproblem des Inpaintings als Differentialgleichung formulieren:

$$(1 - c(\mathbf{x}))Lu - c(\mathbf{x})(u - f) = 0 \tag{2.1}$$

Gleichung 2.1 verbindet dabei zwei verschiedene Kriterien. Der Term $u - f$ beschreibt die Ähnlichkeit zwischen dem Originalbild f und dem Inpaintingresultat u, stellt also ein *Approximationskriterium* dar. Die *Interpolation* hingegen wird durch den Term Lu repräsentiert. Hierbei ist L ein passend gewählter Differentialoperator (z.B. ein Diffusionsoperator wie in Kapitel 3.4 beschrieben).

Die Funktion $c(\mathbf{x}) : \Omega \to [0, 1]$ steuert die Anteile von Approximation und Interpolation am Gesamtverfahren. Diese Gewichtung kann eingesetzt werden, um bei Inpainting die Verlässlichkeit der gegebenen Daten zu berücksichtigen: für $c(\mathbf{x}) = 0$ werden fehlende Daten durch L interpoliert, für $c(\mathbf{x}) = 1$ lösen lediglich Originaldaten die Gleichung.

Für die konkrete Anwendung im Kontext der Bildkompression ist die Verlässlichkeit der Ausgangsdaten klar umrissen. Die Grauwerte der Inpaintingmaske B sind identisch mit denen des Originalbildes f, d.h. es gilt:

$$\forall \mathbf{x} \in B : u(\mathbf{x}) = f(\mathbf{x}) \tag{2.2}$$

Somit ist für alle Punkte der Maske $c(\mathbf{x}) = 1$ zu wählen. Die zu rekonstruierenden Punkte sind gänzlich unbekannt und daher ist $c(\mathbf{x}) = 0$ für alle $\mathbf{x} \in \Omega \setminus B$ ein sinnvoller Wert. Die Verlässlichkeitsfunktion c ist in diesem Fall identisch mit der Indikatorfunktion χ_B und Gleichung 2.1 lässt sich durch Fallunterscheidung vereinfachen:

$$\begin{aligned} \forall \mathbf{x} \in B : \quad & \underbrace{(1 - \chi_B(\mathbf{x}))}_{=0} Lu - \underbrace{\chi_B(\mathbf{x})}_{=1}(u(\mathbf{x}) - f(\mathbf{x})) = u(\mathbf{x}) - f(\mathbf{x}) \overset{2.2}{=} 0 \\ \forall \mathbf{x} \in \Omega \setminus B : \quad & \underbrace{(1 - \chi_B(\mathbf{x}))}_{=1} Lu - \underbrace{\chi_B(x)}_{=0}(u(\mathbf{x}) - f(\mathbf{x})) = Lu \end{aligned}$$

Wendet man den Differentialoperator L mit passenden Randbedingungen für $\partial\Omega$ und den Dirichlet-Randbedingungen 2.2 für B an, so vereinfacht sich die Differentialgleichung 2.1 zu:

$$Lu = 0 \tag{2.3}$$

Bisher ist der Differentialoperator L noch nicht näher spezifiziert worden. Das R-EED-Verfahren von Schmaltz et al. (2010) unterstützt eine Reihe verschiedener Operatoren, allerdings hat sich kantenverstärkende Diffusion aufgrund ihrer Interpolationseigenschaften als beste Wahl erwiesen. Detaillierte Erläuterungen zu Diffusion in besonderem Hinblick auf EED sind in Abschnitt 3.4 zu finden, der neben Grundlagen von Diffusionsprozessen in der Bildverarbeitung die Erweiterung von zweidimensionalem EED auf drei Dimensionen behandelt.

Neben der Wahl des Differentialoperators ist für die Qualität der Rekonstruktion laut Belhachmi et al. (2009) auch die Wahl des Bildbereichs B essentiell. Die Wahl einer optimalen Punktmenge B ist jedoch zum einen nichttrivial und zum anderen muss die Position der bekannten Bildpunkte in der komprimierten Bilddatei gespeichert werden. Daher ist es notwendig, B systematisch zu bestimmen und gleichzeitig im Hinblick auf den Speicherplatzbedarf die Wahl einzuschränken. Wie dies im R-EED-Verfahren umgesetzt wird, ist im folgenden Abschnitt 2.2 beschrieben.

2.2 Adaptive Rechtecksunterteilung

Bisher gibt es für die optimale Wahl der Interpolationsmaske B bei EED-basiertem Inpainting keine theoretischen Ergebnisse, doch Analysen für harmonische Diffusion (Belhachmi et al., 2009) legen nahe, dass die Bedeutung dieser Wahl für die Qualität der Resultate außerordentlich wichtig ist.

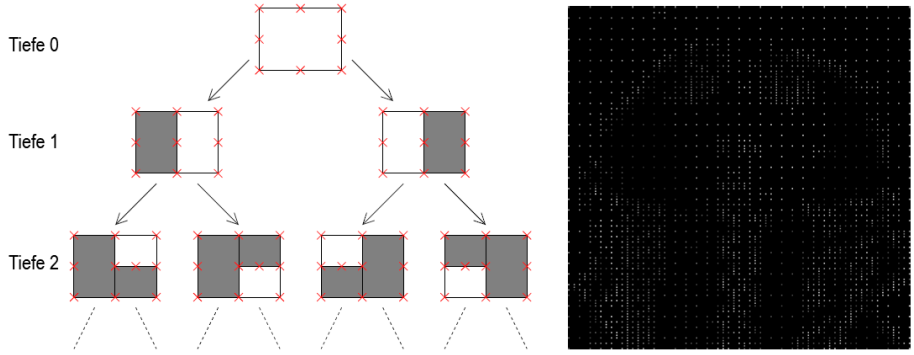


Abbildung 2.1: **Adaptive Rechtecksunterteilung.** Die Abbildung zeigt auf der linken Seite eine schematische Darstellung der Rechtecksunterteilung im R-EED-Verfahren. Das Bild wird sukzessive in Teilbilder unterteilt, wobei jeweils das weiß eingefärbte Bild durch Inpainting rekonstruiert wird und mit der Fehlerschranke T abgeglichen wird. Die rot markierten Bildpunkte werden gespeichert und bilden die Inpaintingmaske, deren Grauwertdarstellung rechts zu sehen ist.

Zwar existiert nur eine endliche Anzahl von Möglichkeiten, n aus m Bildpunkten auszuwählen, für das eingangs in Kapitel 1 erwähnte Beispiel eines Graustufenbildes mit einer Auflösung von 512×512 Pixeln erreichen die Auswahlmöglichkeiten für 10% der Bildpunkte jedoch bereits unüberschaubare Ausmaße ($9,5 \cdot 10^{37006}$ Möglichkeiten).

Zusätzlich stellt sich im Kontext von Bildkompression das Problem, dass neben dem Grauwert der einzelnen Bildpunkte der Interpolationsmaske B auch ihre räumliche Position gespeichert werden muss. Da die Qualität von Inpaintingresultaten zum einen von der Menge der verwendeten Punkte und zum anderen von deren Auswahl abhängt, stehen Speicherplatzbedarf für die räumliche Konfiguration und Anzahl der gespeicherten Punkte in direkter Konkurrenz. Eine suboptimale Punktmaske kann daher insgesamt bessere Kompressionsergebnisse als die Optimalkonfiguration erzielen, wenn sie Platz bei der Speicherung der Positionen einspart und in zusätzliche Punkte investiert. Wählt man die Inpaintingmaske nach einem vorgegebenen, bildunabhängigen Muster, fällt zwar nur der Speicherplatz für die Grauwerte an, die Inpaintingmaske ist jedoch zu unflexibel, um gute Inpaintingergebnisse zu ermöglichen.

Aus diesem Grunde verwenden Schmaltz et al. (2009) im R-EED Verfahren eine Einschränkung der Punktmasken auf ein adaptives Gitter. Es wird zunächst ein festes Punktmuster für rechteckige Bilder festgelegt (z.B. vier Eckpunkte und der Mittelpunkt). Auf der Basis dieser Maske wird ein vorläufiges Inpaintingergebnis berechnet und mit dem Originalbild verglichen. Der so errechnete Fehlerwert wird mit einer Fehlerschranke T verglichen. Ist der Fehler größer als erwünscht, wird das Bild in seiner längsten Seite halbiert. Auf die entstehenden Teilbilder wird rekursiv dieselbe Bildunterteilungsmethode angewendet, bis die Fehlerschranke in allen Teilbildern unterschritten oder eine vorgegebene maximale Rekursionstiefe erreicht wird. Die Fehlerschranke $T := al^d$ kann durch zwei freie Parameter angepasst werden: a definiert die Fehlerschranke für das

(a) (b)

Abbildung 2.2: **Grauwertquantisierung.** Abb. (a) zeigt das Testild Lenna mit seiner ursprünglichen Anzahl von 256 Grauwerten (8 bpp). Reduziert man den Speicherplatzbedarf auf 4 bpp, also 16 Grauwerte, sind deutliche Artefakte auszumachen (Abb. (b)).

Ausgangsbild (Rekursionstiefe 0) und l steuert den Grad der Anpassung an die Rekursionstiefe d.

Diese Methode ermöglicht es, Speicherplatz durch vorgegebene Punktmuster zu sparen, gleichzeitig jedoch eine Anpassung an die lokalen Gegebenheiten verschiedener Bildregionen vorzunehmen: in semantisch wichtigen Bildregionen (z.B. Kanten) die schwieriger zu rekonstruieren sind, speichert das Unterteilungsverfahren mehr Punkte als im Rest des Bildes. Die einzigen Informationen, die ein Decodierer für eine vollständige Rekonstruktion der Inpaintingmaske benötigt, sind die gespeicherten Grauwerte und die erfolgten Bildunterteilungen. Letztere lassen sich effizient durch einen binären Baum darstellen (siehe Abbildung 2.1).

2.3 Grauwertquantisierung

Eine klassische Form der verlustbehafteten Kompression ist die Vergröberung der Diskretisierung des Grauwertbereichs, im Weiteren als *Quantisierung* bezeichnet. Für Graustufenbilder ist es üblich, Grauwerte der Größe eines Bytes zu benutzen, d.h. eine Quantisierung des kontinuierlichen Grauwertbereichs durch $256 = 2^8$ diskrete Grauwerte wird vorgenommen.

Das menschliche Auge kann jedoch nur eine begrenzte Anzahl von Grauwerten unterscheiden, daher bietet sich eine Verringerung der Grauwertanzahl für wahrnehmungsorientierte Kompression an (Tönnies, 2005). Verringert man beispielsweise die Anzahl der Grauwerte von 2^8 auf $2^6 = 64$, sind durch visuelle Wahrnehmung kaum Unterschiede festzustellen. Um die Zahlen des Grauwertbereiches binär darstellen zu können, werden somit nur noch 6 bit statt der ursprünglichen 8 bit benötigt. Damit werden allein durch die neuerliche Quantisierung gegenüber dem unkomprimierten Originalbild 2 bit pro Bildpunkt ein-

gespart. Die Verringerung der Grauwertanzahl ist jedoch nicht nur bei einer derartigen naiven Codierung von Vorteil, auch Entropie-Codierungen (siehe Abschnitt 2.4) profitieren davon.

Bei der Reduktion von n Grauwerten auf q Grauwerte ($n > q$) durch gleichmäßige Quantisierung wird hierbei der Grauwertbereich in m Intervalle der Länge n/q partitioniert. Dadurch ergibt sich eine Abbildung $m_q(g)$ (*quantisation map*) der ursprünglichen Grauwerte $G_n = \{0, \ldots, n-1\}$ auf den neuen Grauwertbereich $G_q = \{0, \ldots, q-1\}$, die jedem Wert aus G_n die entsprechende Intervallnummer aus G_q zuordnet.

Jedem der quantisierten Grauwerte aus G_q kann ein *Rekonstruktionswert* (Strutz, 2002) aus dem ursprünglichen Grauwertbereich zugeordnet werden. Durch diese Abbildung können jedoch keine Informationen, die bei der Reduktion auf die Grauwertmenge G_n verloren wurden, wiederhergestellt werden. Die Rekonstruktionszuordnung wird durch eine weitere quantisation map m_n dargestellt:

$$m_n : G_q \to G_n : g(x) \mapsto \left\lfloor \frac{xn}{q} + \frac{1}{2} \right\rfloor$$

Im Kontext des R-EED-Verfahrens ist zu beachten, dass die Grauwerte im Komprimierungsvorgang nach der Rechtecksunterteilung quantisiert werden und bei der Dekomprimierung vor dem Inpainting rekonstruiert werden. Durch den Inpaintingprozess auf den rekonstruierten Grauwerten werden auch Grauwerte aus G_n angenommen, die zwischen den durch die Rekonstruktionsabbildung m_n gewonnenen Werten liegen. Da die Parameter des Inpaintingprozesses so optimiert werden, dass das Resultat dem Originalbild möglichst ähnlich ist, kann dadurch der durch Quantisierung erfolgte Informationsverlust gering gehalten werden.

2.4 Entropie-Codierung

Die Grauwerte der anhand der Inpaintingmaske B ausgewählten Bildpunkte können nach der Quantisierung ohne weiteren Informationsverlust durch sogenannte Entropie-Codierung gespeichert werden. Diese Kompressionsmethoden betrachten die Grauwerte einzelner Pixel als Werte, die möglichst speichereffizient sequentiell übertragen werden sollen. Daher spricht man in diesem Zusammenhang auch von der Ausnutzung von *Signalredundanz* (Tönnies, 2005).

Um eine verlustlose Übertragung aller Grauwerte zu garantieren, muss jeder Wert durch ein eindeutiges Symbol, das *Codewort*, repräsentiert werden. Bei digitalen Übertragungen definieren sich Codewörter über den *Codewert*, eine Zahl in Binärdarstellung, und die *Codelänge*, welche in der Einheit *bit* gemessen wird (Strutz, 2002). In unkomprimierten Bildern werden alle Grauwerte durch Zahlencodes gleicher Länge repräsentiert (z.B. 256 Grauwerte durch Codes der Länge 8 bit). In der Regel sind in einem Bild jedoch nicht alle Grauwerte in gleicher Anzahl vertreten. Betrachtet man die Auftretenswahrscheinlichkeit p_i eines jeden Grauwertes in einer Folge statistisch unabhängiger Codewörter, so lässt sich deren mittlerer Informationsgehalt berechnen:

$$H = \sum_{i=1}^{K} p_i \log_2 \frac{1}{p_i} \tag{2.4}$$

H wird als *Entropie* bezeichnet und gibt eine untere Schranke für die mittlere Codewortlänge aller Codierungsverfahren vor. Im Kontext der Bildkompression bedeutet dies: Sind Grauwerte im Bild unterschiedlich häufig vertreten, so kann eine Reduzierung des Speicherplatzbedarfs erreicht werden, indem häufig auftretende Grauwerte durch kurze Codewörter und wenig auftretende Grauwerte durch lange Codewörter repräsentiert werden. Ziel von Entropie-Codierungs-Methoden ist es demnach, die mittlere Codewortlänge der Entropie anzunähern.

Im Folgenden werden in aller Kürze die in R-EED eingesetzten Entropie-Verfahren beschrieben.

Huffman-Codierung Eine im Sinne der Entropie optimale Codierung wird erreicht, indem Codes über einen binären Baum zugewiesen werden. Grauwerte werden derart in den Knoten des Baumes gespeichert, dass die Häufigkeit mit zunehmender Baumtiefe abnimmt. Die Codes werden erzeugt, indem ausgehend von der Wurzel jeder Verzweigung des Baumes eindeutig eine 1 oder eine 0 zugeordnet wird. Um eine Decodierung zu ermöglichen, muss der Baum mit übertragen werden (Huffman, 1952).

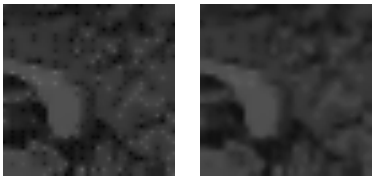
Bereichscodierung Bereichscodierung basiert auf dem Ansatz, ein Codewort für das gesamte zu codierende Signal zu verwenden, statt jedem Symbol ein eigenes Codewort zuzuweisen. Es wird zunächst ein beliebiges Intervall $[a, b]$ definiert, das alle möglichen Zahl-Codewörter enthält. Für jedes Symbol des Signals wird ein Teilintervall definiert, dessen Länge die jeweilige Vorkommenshäufigkeit repräsentiert. Das zum ersten Symbol der Signalsequenz gehörige Teilintervall wird auf gleiche Weise weiter unterteilt. Auf diese Art wird mit jedem übertragenen Symbol das Intervall eingeschränkt. Als Codewort wird ein beliebiger Wert aus dem letzten Intervall verwendet. Zur Decodierung müssen die Länge des Signals und die Symbolhäufigkeiten bekannt sein (Martin, 1979).

LZW-Codierung Das Codierungsverfahren von Welch (1984) stellt keine reine Entropie-Codierung dar, sondern beruht auf dem Prinzip der *Präcodierung*. Mehrere Codewörter werden zu Phrasen zusammengefasst, welche in einem sogenannten Wörterbuch gespeichert werden. Auf diese Art können gleiche Abfolgen von n gleichen Phrasen identifiziert werden und durch ein kurzes Wörterbuch-Symbol und die Zahl n codiert werden (*Lauflängencodierung*).

gzip verwendet eine Kombination aus Huffman-Codierung und dem Präcodierungsverfahren LZ77 (Ziv und Lempel, 1977), einem Vorgänger der im LZW-Verfahren eingesetzten Präcodierung.

bzip2 Kernstück des bzip2-Verfahrens ist ein Blocksortierungsalgorithmus (Burrows und Wheeler, 1994), der für sich wiederholende Symbolsequenzen sorgt. Diese Sortierung wird durch Lauflängen- und Huffman-Codierung ausgenutzt.

PAQ ist ein komplexes *context-mixing*-Verfahren, welches Prädikationsmethoden mit neuralen Netzwerken und Bereichscodierung kombiniert. In dieser Arbeit wird eine modifizierte Version von PAQ8 eingesetzt (paq8o8z-feb29, Mahoney (2005, 2010)).



(a) (b)

Abbildung 2.3: **Umkehrung der Inpaintingmaske.** Die obigen Abbildungen demonstrieren den Effekt der Umkehrung der Inpaintingmaske anhand eines Ausschnitts aus einem medizinischen Testbild. In Abb. (a) heben sich isolierte, durch Grauwertoptimierung veränderte Maskenpunkte deutlich vom Restbild ab. Dieser Effekt wird durch Maskeninversion behoben (Abb. (b)).

In den Experimenten von Schmaltz et al. (2010) haben sich PAQ und Bereichscodierung als die beiden effektivsten Verfahren für den Einsatz im R-EED-Verfahren erwiesen.

2.5 Grauwertoptimierung

Bei einer fest vorgegebenen Menge bekannter Punkte hängt das Resultat des Inpaintingschrittes nur von den Inpainting-Parametern und den Grauwerten der bekannten Punkte ab. Modifiziert man die Grauwerte, ändert sich auch das Ergebnis des Inpaintings. Daher bietet sich hier ein Ansatzpunkt, dem optimalen Rekonstruktionsergebnis näher zu kommen, ohne die Position gespeicherter Punkte zu verändern. Zusätzlich motivieren lässt sich eine Veränderung der Grauwerte durch Bildrauschen im Originalbild: Da die Punktmuster der Quaderunterteilung fest vorgegeben sind, kann es vorkommen, dass Bildinformationen aus verrauschten Bildpunkten rekonstruiert werden. Durch Anpassung des Grauwertes kann dies verhindert werden.

Der R-EED Algorithmus führt Grauwertoptimierung sequenziell für einzelne Punkte der Inpaintingmaske durch. Der Grauwert jedes einzelnen Punktes wird nach oben oder unten korrigiert und die Korrektur bei einer Verbesserung des Inpaintingresultats beibehalten. Da die Inpaintingqualität auch von der Parameterwahl abhängt, werden abwechselnd Grauwerte und Inpainting-Parameter optimiert, bis keine Verbesserung mehr erreicht werden kann.

2.6 Umkehrung der Inpaintingmaske

Beschränkt man R-EED auf adaptive Rechtecksunterteilung, Inpainting und Entropie-Codierung, so stellt das Verwerfen der Bildpunkte des Inpaintingbereichs $\Omega \setminus B$ den einzigen verlustbehafteten Kompressionsschritt dar. In diesem Fall sind insbesondere auch die gespeicherten Grauwerte der Interpolationsmaske B identisch mit denen des Originalbildes.

Setzt man jedoch die verlustbehaftete Grauwertquantisierung (Abschnitt 2.3) und/oder Grauwertoptimierung (Abschnitt 2.5) ein, so ist es möglich, dass die Werte der Interpolationsmaske von den Originalwerten abweichen. Der Inpaintingprozess aus Abschnitt 2.1 nimmt keinerlei Änderungen an den Bild-

punkten des Inpaintingbereichs vor, daher bleibt der dort vorliegende Fehler unbeeinträchtigt erhalten.

Um einen zusätzlichen Qualitätsgewinn zu erreichen, setzen Schmaltz et al. (2010), ähnlich wie Bae und Weickert (2010), eine an Hopscotch-Methoden (Gourlay, 1970) orientierte Umkehrung der Inpaintingmaske in Kombination mit einem zusätzlichen Inpaintingschritt ein. Dabei werden die Rollen von Inpaintingbereich und Bildbereich vertauscht. Der neue Inpaintingbereich I_H besteht aus der Vereinigung kugelförmiger Bereiche mit Radius r_H um alle Punkte der alten Inpaintingmaske B. Die neue Inpaintingmaske B_H ergibt sich als $B_H := \Omega \setminus I_H$.

Auf diese Weise werden zwei verschiedene Effekte erzielt: einerseits kann den Fehlern aus verlustbehafteten Kompressionsschritten entgegen gewirkt werden, andererseits wird eine Verbesserung der wahrnehmungsbasierten Qualität ermöglicht. Die Punkte der Interpolationsmaske heben sich meist deutlich von den durch Inpainting rekonstruierten Bildteilen ab. Insbesonders durch Grauwertoptimierung wird dieser Umstand noch zusätzlich verstärkt. Durch ein neuerliches Inpainting mit invertierter Inpaintingmaske wird dieser Effekt vermieden (siehe Abb. 2.3).

Aus Gründen der Lesbarkeit wird im Folgenden für den in diesem Abschnitt beschriebenden Kompressionschritt die verkürzte Bezeichnung Maskeninversion eingesetzt.

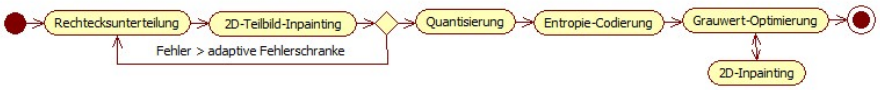
2.7 Gesamtverfahren

2.7.1 Zusammenfassung

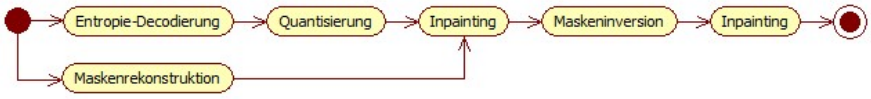
Der Kern und erste Schritt des R-EED-Codierers ist die adaptive Rechtecksunterteilung mit Teilbild-Inpainting. Hierbei wird zwischen Unterteilungs- und Inpaintingschritten abgewechselt, bis die Fehlerschranke T in allen Teilbildern unterschritten wird oder die maximale Rekursionstiefe erreicht ist. Die Bildunterteilung wird in einen binären Baum codiert. Anhand eines vorgegebenen Punktmusters p werden Grauwerte eines jeden Teilbildes in eine geordnete Liste überführt (nach Auftreten von links oben nach rechts unten im Bild).

Ist auf diese Weise die Inpaintingmaske bestimmt, werden die Grauwerte der Liste zunächst einer Quantisierung auf $q \leq 256$ Grauwerte unterzogen. Im Anschluss wird die Grauwertliste durch Entropie-Codierung verlustfrei komprimiert. Die Schritte der Grauwertoptimierung und Parameterfindung für die Maskeninversion können entweder vor der Entropie-Codierung oder nachträglich durchgeführt werden, indem das komprimierte Bild decodiert und dann modizifiert wird. Alle verwendeten Parameter, der binärcodierte Baum und die komprimierte Grauwertliste werden im Dateikopf der komprimierten Bilddatei gespeichert.

Der Decodierer muss zunächst anhand des Baumes und des Punktmusters p die räumliche Konfiguration der Inpaintingmaske rekonstruieren. Davon unabhängig ist die Dekomprimierung der Grauwertliste durch Entropie-Codierung und anschließende Quantisierung auf 256 Grauwerte. Danach erfolgt das Inpainting unter Verwendung der im Dateikopf gespeicherten Parameter. Abschließend kann eine Maskeninversion erfolgen.



(a)



(b)

Abbildung 2.4: **UML-Darstellung des R-EED-Verfahrens.** Der Kompressionsvorgang, dargestellt in Abb. (a) verläuft streng sequentiell. Jeder der Kompressionsschritte beruht auf den Ergebnissen des vorherigen. Bei der Dekompression (Abb. (b)) sind die Rekonstruktion der Inpaintingmakse aus dem gespeicherten Binärbaum und die Entropie-Decodierung voneinander unabhängig.

2.7.2 Parameteroptimierung

Im vorherigen Abschnitt ist der Kompressionsalgorithmus unter der Annahme beschrieben, dass alle Parameter fest vorgegeben sind. Um eine gute Qualität bei hoher Kompressionsrate zu erhalten, müssen die Parameter jedoch an das Bild angepasst werden. Im folgenden wird eine Übersicht über alle in diesem Kapitel eingeführten R-EED-Parameter gegeben.

Rechtecksunterteilung: Die Rechtecksunterteilung benötigt drei Parameter: a, l und p. a gibt die Fehlerschranke für die Baumtiefe 0 vor, während l die Anpassung an die aktuelle Tiefe d über die Formel $T := al^d$ bestimmt. Zusätzlich muss das verwendete Punktmuster p bekannt sein.

Inpainting: Die verwendete EED-Methode hat zwei freie Parameter: einen Kontrastparameter λ und einen Glättungsparameter σ (siehe Abschnitt 3.4). Da die Wahl von σ jedoch nur geringe Auswirkungen auf die Ergebnisse hat, wird der Wert von Schmaltz et al. (2010) auf $\sigma = 0.8$ fixiert.

Quantisierung: Die Zahl $q \leq 256$ der neu quantisierten Grauwerte ist der einzige Parameter dieses Schrittes.

Entropie-Codierung: Hier muss lediglich festgelegt werden, welche Codierung benutzt wird.

Maskeninversion: Der Radius r_H des Kreises um die Punkte der Interpolationsmaske muss festgelegt werden.

Die Optimierung aller vorgenannten Parameter ist ein langwieriger Prozess, da die Parameter eines Kompressionsschrittes auch Einfluss auf die Resultate anderer Kompressionsschritte haben können. Bis auf die Wahl der Entropie-Codierung, die lediglich die Dateigröße beeinflusst, und den Radius der Maskeninversion stehen alle andere Parameter in gegenseitiger Wechselwirkung und müssen ganzheitlich optimiert werden.

In der konkreten Implementierung kann Rechenzeit gespart werden, indem die Reihenfolge der Parameteroptimierungen geschickt gewählt wird. Der Rekonstruktionsfehler von Teilbildern wird nur durch den Kontrastparameter λ und die Wahl p des Punktmusters beeinflusst. Daher sollten zunächst für gleichbleibendes λ und p alle anderen Parameter optimiert werden, um eine Neuberechnung der Inpaintingschritte zu vermeiden.

Kapitel 3

Grundlagen des C-EED-Verfahrens

In diesem Kapitel werden die theoretischen Grundlagen des C-EED-Verfahrens und seine Unterschiede zu der zugrundeliegenden R-EED-Methode (Schmaltz et al., 2009) erläutert. Hierzu werden zunächst in Abschnitt 3.1 die Redundanzen in dreidimensionalen Bildern diskutiert, die C-EED einsetzt, um gegenüber R-EED einen qualitativen Vorteil zu erreichen.

3.1 Redundanzen und Irrelevanzen in dreidimensionalen Bildern

Wendet man das R-EED Verfahren schnittbildweise auf ein dreidimensionales Bild an, so ergeben sich drei verschiedene Quellen von Redundanz:

Dateikopf-Redundanz: Im Dateikopf der gespeicherten Dateien sind eine Reihe von Informationen mehrfach enthalten, so zum Beispiel die Bildgröße. Die maximal erreichbare Einsparung ist ein voller Dateikopf pro Schnittbild. Bei hoch aufgelösten Bildern hat die Größe des Dateikopfes allerdings nur einen geringen Anteil an der Gesamtgröße der Datei, daher sind nur geringe bis moderate Platzgewinne zu erwarten.

Codierungs-Redundanz: Nutzt man Entropie-Codierung auf den Grauwerten einer einzelnen Inpaintingmaske, ergibt sich je nach Entropie-Verfahren auf unterschiedliche Art und Weise ein erhöhtes Kompressionspotential. So können Codewörter nun einheitlich für das ganze Bild vergeben werden, was redundante Huffmann-Bäume oder Wörterbücher für jedes einzelne Schnittbild überflüssig macht. Diese Codes können dann auch anhand der globalen Vorkommenshäufigkeiten im ganzen 3D-Bild gewählt werden, statt nur auf lokale Schnittbildhäufigkeiten optimiert zu werden. Weiterhin erhöht sich die Anzahl sich wiederholender Bitmuster, insbesondere da die Schnittbilder in der Regel Ähnlichkeiten in der Bildstruktur aufweisen.

Inpainting-Redundanz: Im zweidimensionalen Fall nutzt Inpainting die Ähnlichkeit der Nachbarpunkte in x- und y-Dimension, um durch Transport-

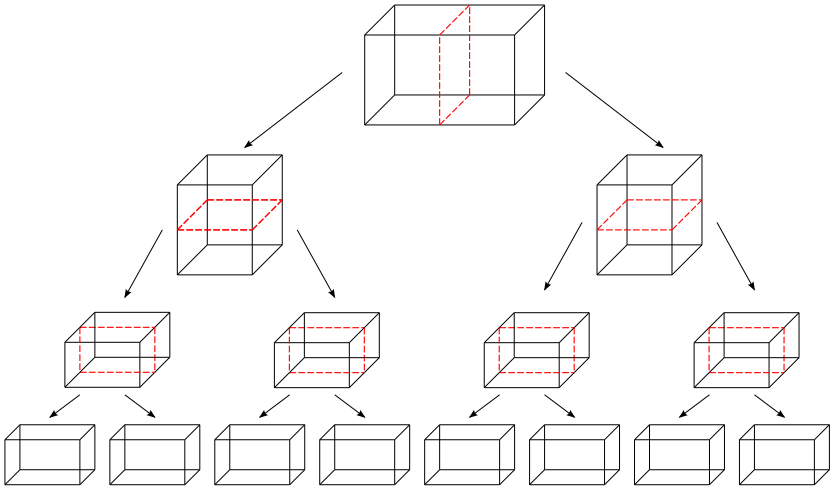


Abbildung 3.1: **Quaderunterteilung.** Im obigen Beispiel wird ein Quader durch Halbieren in der jeweils größten Dimension in Teilquader zerlegt. In analoger Weise wird durch Quaderunterteilung ein 3D-Bild zerlegt. Diese Abbildung zeigt eine Zerlegung bis zu einer Baumtiefe von drei, in der alle möglichen Unterteilungen vorgenommen werden.

gleichungen Grauwerte aus bekannten Bildregionen in unbekannte Bildregionen zu überführen und somit die fehlenden Teile zu rekonstruieren. Dabei werden allerdings nur die zwei genannten Bilddimensionen ausgenutzt. In 3D-Bildern liegt jedoch in der Regel auch in der z-Dimension ein Zusammenhang zwischen Nachbarpixeln vor. Die dritte Dimension ist nicht zwangsläufig gleichberechtigt, da die Abtastrate bei bildgebenden Verfahren unter Umständen in einer Dimension geringer ausfällt und bei Videos oft eine geringe Abtastrate in der Zeitdimension und Diskontinuitäten durch Videoschnitt gegeben sind. Dennoch sind auch in der dritten Dimension Redundanzen vorhanden, die durch dreidimensionales Inpainting ausgenutzt werden können, um mit geringerer Anzahl von Punkten die Gleiche durchschnittliche Rekonstruktionsqualität wie zweidimensionales Schnittbild-Inpainting zu erreichen.

In den folgenden Abschnitten werden Möglichkeiten diskutiert, die genannten Redundanzen auszunutzen, um bei gleicher Kompressionsrate eine höhere Qualität zu erreichen.

3.2 Adaptive Quaderunterteilung

Das R-EED-Verfahren führt auf Eingabebildern eine Rechtecksunterteilung durch, um so die Interpolationsmaske aus einem adaptiven Gitter zu konstruieren (siehe Abschnitt 2.2). Dabei werden rechteckige Bilder in der längsten Seite halbiert, woraus zwei rechteckige Teilbilder entstehen. Überträgt man dieses Prinzip auf dreidimensionale Bilder, so erhält man durch Halbieren in der größten der drei Quaderdimensionen zwei quaderförmige Bilder.

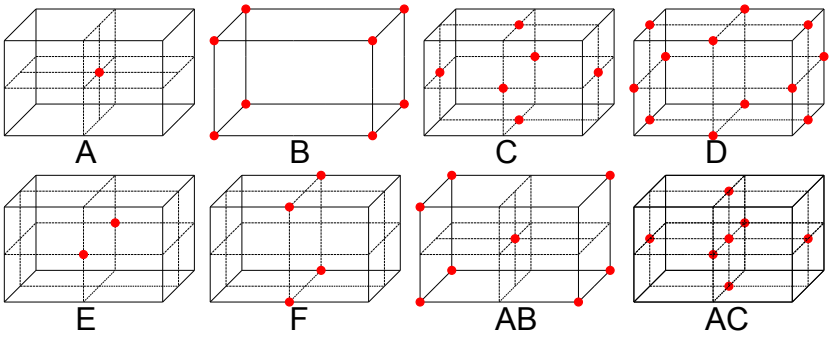


Abbildung 3.2: **Punktmuster.** Das adaptive Gitter der Quaderunterteilung setzt sich aus Punktmustern zusammen, die auf Teilbilder angewendet werden. Die Muster A-E bilden Grundbausteine, aus denen in beliebiger Art und Weise weitere Muster zusammengesetzt werden können. AB und AC haben sich in Experimenten als erfolgreichste Musterkombinationen erwiesen.

Da weiterhin das Bild halbiert wird, also in jedem Schritt zwei neue Teilbilder entstehen, lässt sich die Unterteilung wie im zweidimensionalen Fall durch einen Binärbaum repräsentieren (siehe Abb. 3.1). Die Punktmuster, aus denen sich das adaptive Gitter zusammensetzt, werden nun in Relation zu den dreidimensionalen Quadern der Teilbilder definiert. Hier ergeben sich durch die Dreidimensionalität mehr sinnvolle Optionen als im zweidimensionalen Fall.

In R-EED werden die Punktmuster durch verschiedene Kombinationen der vier Ecken der rechteckigen Teilbilder, deren vier Seitenmittelpunkte und den Rechtecks-Mittelpunkt definiert. In Anlehnung an den zweidimensionalen Fall werden als Grundbausteine für Punktmuster im Dreidimensionalen die acht Ecken des Quaders, die sechs Mittelpunkte der Seitenflächen, die zwölf Kantenmittelpunkte, sowie der Mittelpunkt gewählt (siehe Abb. 3.2).

Beliebig viele dieser Grundbausteine können zu neuen Mustern kombiniert werden, allerdings sind nicht alle Konfigurationen im Kontext der Bildkompression sinnvoll. Zunächst ist zu beachten, dass einige der Kombinationen redundant sind. So kann Beispielsweise ein Muster, welches Mittelpunkt, Ecken und Seitenflächenmittelpunkte des Quaders enthält, auch durch eine Unterteilung in vier Teilbilder erreicht werden, deren Eckpunkte gespeichert werden. Grundsätzlich sind Punktmuster mit weniger Bildpunkten pro Teilbild flexibler als Muster mit vielen Teilpunkten, da sie eine feinere Regulierung der Punktanzahl in einzelnen Bildregionen erlauben.

Darüber hinaus spielt die Verteilung der Maskenpunkte im Bild eine Rolle für die Rekonstruktionsqualität und beeinflusst auch die Baumtiefe, die benötigt wird, um die Fehlerschranke T zu unterschreiten. Für verschiedene Bilder können unterschiedliche Punktmuster optimale Ergebnisse liefern. In dieser Arbeit wird die Wahl der Punktmuster auf die Kombination von bis zu drei Grundbausteinen reduziert, wobei bei mehreren Grundbausteinen der Quadermittelpunkt immer hinzugenommen wird. Untersuchungen zu den Auswirkungen verschiedener Punktmuster auf die Kompressionsrate und Qualität sind in Kapitel 5 zu finden. In Anlehnung an die Ergebnisse der 2D-Punktmusteruntersuchung von Schmaltz et al. (2010) sind gute Ergebnisse vor allem mit den Punktmustern AB, AC und AD zu erwarten.

Verfahren, die auf einer adaptiven Quaderunterteilung (engl. cuboidal subdivision) in Kombination mit kantenverstärkender Diffusion (engl. edge-enhancing diffusion) beruhen, werden im Folgenden als C-EED-Verfahren bezeichnet.

3.3 C-EED2D

Mit Hilfe der adaptiven Quaderunterteilung aus Abschnitt 3.2 kann ein Verfahren definiert werden, welches weiterhin im Inpaintingschritt auf zweidimensionale Diffusion zurückgreift und gleichzeitig die Dateikopf- und Entropie-Redundanzen dreidimensionaler Bilder ausnutzt.

Die dreidimensionale Aufteilung des Bildes erfolgt daher bis zur Tiefe $\log_2(n_z)$ zunächst nur in z-Richtung. In dieser Tiefe wurden die Teilbilder in der dritten Dimension auf eine Länge von eins reduziert, d.h. sie entsprechen zweidimensionalen Schnittbildern in der x-y-Ebene. Die weitere Bildunterteilung erfolgt wie im zweidimensionalen Fall durch Halbierung der längeren Rechteckseite. Der Binärbaum der adaptiven Quaderunterteilung vereint somit alle Bäume der adaptiven Rechtecksunterteilung der Schnittbilder, ohne die Anpassungsfähigkeit der adaptiven Gitter einzuschränken.

Die restlichen Kompressions- und Dekompressionsschritte des R-EED-Algorithmus (siehe Abb. 2.4) bleiben in ihrer grundsätzlichen Form und Abfolge erhalten und operieren wie gewohnt auf der Grauwertliste, die sich aus der Inpaintingmaske ergibt. Der Inpaintingschritt erfolgt durch zweidimensionales EED-Inpainting auf den einzelnen Schnittbildern. Quantisierung, Grauwertoptimierung und Interpolationsumkehrung profitieren nicht von der gemeinsamen Interpolationsmaske für alle Schnittbilder, Entropie-Codierung kann jedoch die in Abschnitt 3.1 beschriebenen Redundanzen ausnutzen.

Grundsätzlich ist es möglich, alle freien Parameter des Algorithmus einzeln für jedes Schnittbild festzulegen, wie es bei einer Anwendung der R-EED Methode auf Einzelbildern der Fall ist. Für C-EED2D wird allerdings eine globale Optimierung vorgenommen, d.h. die Parameter werden schnittbildübergreifend für das gesamte 3D-Bild festgelegt. Dadurch reduziert sich auch die Dateikopf-Größe.

3.4 Diffusion

Um Redundanzen in der dritten Dimension ausnutzen zu können, wird im C-EED-Verfahren Inpainting unter Verwendung dreidimensionaler Diffusion eingesetzt.

3.4.1 Physikalische Grundlagen

Der Begriff Diffusion bezeichnet den physikalischen Vorgang des Konzentrationsausgleiches zwischen verschiedenen Stoffen. Als klassisches Beispiel für diesen Prozess wird in Crank (1975) die Diffusion von Jodlösung und klarem Wasser genannt. Befinden sich beide Stoffe in einem abgeschlossenen System, beispielsweise in einem zylindrischen Behältnis, so besteht zunächst eine klar erkennbare Trennung zwischen den Stoffen. Im zeitlichen Verlauf erfolgt jedoch eine Vermischung, die schließlich in einem stabilen Gleichgewicht mit einer gleichmäßigen Verteilung von Jodmolekülen in der gesamten Flüssigkeit endet.

Derartige Diffusionsvorgänge basieren zwar auf zufälligen Teilchenbewegungen, die nicht vorhergesagt werden können, der Gesamtprozess lässt sich mathematisch jedoch analog zur Hitzeleitung formulieren, wenn man betrachtet, wie viele Teilchen sich durch eine senkrecht zur Diffusionsrichtung stehende Flächeneinheit bewegen (siehe auch Carslaw und Jaeger, 1959). Das Ficksche Gesetz besagt, dass sich der Teilchenfluss $\mathbf{j} \in \mathbb{R}^n$ in Diffusionsprozessen proportional zu dem Konzentrationsgradienten $\nabla \mathcal{C} = (\partial_1 \mathcal{C}, \ldots, \partial_n \mathcal{C})$ verhält:

$$\mathbf{j} = -D \nabla \mathcal{C} \tag{3.1}$$

Die symmetrische Matrix $D \in \mathbb{R}^{n \times n}$ wird hierbei als Diffusionstensor bezeichnet und variiert je nach Diffusionsprozess in ihren Eigenschaften. In isotropen Medien, in denen der Diffusionsprozess gleichmäßig in alle Richtungen erfolgt (z.B. Flüssigkeiten), hat die Matrix nur einen Eigenwert, d.h. D kann auch durch einen Skalarwert dargestellt werden. In anderen Medien wie Kristallen erfolgt die Diffusion anisotrop, also richtungsabhängig unterschiedlich stark.

Neben anisotroper und isotroper Diffusion wird auch lineare und nichtlineare Diffusion unterschieden: bei nichtlinearer Diffusion ist der Diffusionstensor $D(\mathcal{C})$ abhängig von der Konzentration, im linearen Fall kann D als Konstante aufgefasst werden.

Darüber hinaus gilt in einem geschlossenen System das Gesetz der Masseerhaltung, welches durch die folgende Kontinuitätsgleichung beschrieben wird:

$$\partial_t \mathcal{C} = -\mathrm{div} J \tag{3.2}$$

Mit Hilfe der Gleichung 3.2 kann das Ficksche Gesetz 3.1 in einen zeitlichen Zusammenhang gebracht werden. Es ergibt sich die Diffusionsgleichung:

$$\partial_t \mathcal{C} = \mathrm{div}(D \nabla \mathcal{C}) \tag{3.3}$$

3.4.2 Diffusion in der Bildverarbeitung

Für die Zwecke der Bildverarbeitung wird das physikalische Konzept auf die Grauwerte diskreter Rasterbilder adaptiert. Das Bild ist hierbei das abgeschlossene System, in dem ein Konzentrationsausgleich zwischen Grauwerten stattfindet. Die zeitliche Evolution eines Bildes $u(\mathbf{x}, t)$, $\mathbf{x} \in \mathbb{R}^n$ wird über die künstliche Zeitvariable $t \in [0, \infty)$ dargestellt. Am Ende dieses Prozesses steht, wie im physikalischen Vorbild, ein Gleichgewichtszustand. Für $t \to \infty$ nehmen alle Bildpunkte von u den mittleren Grauwert des Originalbildes $f(\mathbf{x}) \equiv u(\mathbf{x}, 0)$ an.

Eine vollständige mathematische Formulierung dieses Prozesses ist durch das folgende *Anfangswertproblem* (AWP) gegeben:

$$\partial_t u = div(D \nabla u) \quad \text{auf} \quad \Omega \times (0, \infty), \tag{3.4}$$

$$u(\mathbf{x}, 0) = f(\mathbf{x}) \quad \text{auf} \quad \Omega, \tag{3.5}$$

$$\langle D \nabla u, \mathbf{n} \rangle = 0 \quad \text{auf} \quad \partial\Omega \times (0, \infty) \tag{3.6}$$

Zusätzlich zur Diffusionsgleichung 3.4 wird hierbei noch das Originalbild f in Gleichung 3.5 als Anfangswert des Bildes in der zeitlichen Entwicklung definiert. Gleichung 3.6 spezifiziert die Randbedingungen des AWP. Hierbei bezeichnet $\mathbf{n}$ den äußeren Normalenvektor des jeweiligen Randpunktes, sowie $\langle \cdot, \cdot \rangle$ das euklidische Skalarprodukt.

Wie im physikalischen Vorbild wird im Kontext der Bildverarbeitung zwischen isotropen und anisotropen, sowie zwischen linearen und nichtlinearen Diffusionsprozessen unterschieden.

3.4.3 Skalenraumeigenschaften und Gutgestelltheit

Die Anwendung linearer Diffusion auf ein Bild ist äquivalent zur Faltung mit einer Gauß-Funktion der Form:

$$K_\sigma(\mathbf{x}) := \frac{1}{2\pi\sigma^2} \exp\left(\frac{|\mathbf{x}|^2}{2\sigma^2}\right) \tag{3.7}$$

Ein derartiger Gauß-Filter glättet das Eingabebild. Feine Bilddetails und insbesondere Bildrauschen verschwinden in der zeitlichen Entwicklung zuerst. Gröbere Strukturen bleiben länger erhalten, verschwimmen jedoch im Zuge der Annäherung an den Gleichgewichtszustand.

Die gefilterten Bilder weisen dabei Eigenschaften auf, die für eine Verwendung in der Bildverarbeitung von Vorteil sind. So existiert eine eindeutige Lösung der linearen Diffusionsgleichung, die stetig vom Ausgangsbild abhängt, d.h. das korrespondierende Anfangswertproblem ist *gutgestellt*. Weiterhin wird der mittlere Grauwert des Bildes erhalten und die maximalen und minimalen Grauwerte werden nicht unterschritten (*Maximum-Minimum-Prinzip*). Zudem sind die Diffusionsergebnisse translations- und rotationsinvariant.

Diese und weitere Kriterien werden in *Skalenraumkonzepten* zusammengefasst. Ausgehend von der Annahme, dass die menschliche visuelle Wahrnehmung auf Detailhierarchien beruht, definieren Skalenräume durch die sukzessive Anwendung von Filtern wie linearer Diffusion ebensolche Hierarchien. Die konkreten Anwendungsmöglichkeiten für Skalenräume in Gebieten der Bildverarbeitung sind vielfältig. So kann beispielsweise Kantendetektion oder Segmentierung bessere Ergebnisse erzielen, wenn sie auf verschiedenen Skalen des Gauß-Skalenraumes operiert (vgl. Tönnies, 2005). Der lineare Skalenraum oder Gauß-Skalenraum ist dabei ein klassisches Konzept, das sich auch auf nichtlineare isotrope und anisotrope Diffusionsmethoden übertragen lässt.

Es ist allerdings zu beachten, dass sich Eigenschaften, die im kontinuierlichen Modell vorliegen, nicht zwangsläufig auf diskrete Implementierungen, wie die in Kapitel 4 genannten, übertragen. Daher ginge eine ausführlichere Diskussion von Skalenraumeigenschaften und Gutgestelltheit kontinuierlicher Diffusionsmethoden nicht nur über den Rahmen dieser Arbeit hinaus, sondern wäre auch dem Hauptgegenstand wenig dienlich. Eingehende theoretische Betrachtungen für alle bisher genannten Arten von Diffusion sind in "Anisotropic Diffusion in Image Processing" von Weickert (1996a) zu finden.

3.4.4 Kantenverstärkende Diffusion

Zweidimensionale Ausgangssituation

Kantenverstärkende Diffusion (EED, engl. edge-enhancing diffusion) ist ein nichtlinearer, anisotroper Diffusionsprozess, der eingesetzt wird, um Schwachstellen linearer und nichtlinearer, isotroper Diffusion zu beheben. Lineare Diffusion bzw. Gaußglättung hat den Nachteil, dass Kanten, die semantisch wichtige Bildinformationen enthalten, verschwimmen und ihre Position nicht mehr nachzuvollzie-

hen ist. Nichtlinear-isotrope Filter (Perona und Malik, 1987) können Kanten erhalten, indem kantennahe Diffusion verringert oder unterbunden wird. Dadurch entfallen an den Kanten allerdings auch die positiven Effekte der homogenen Diffusion (d.h. z.B. Bildrauschen bleibt nahe der Kanten erhalten).

Um dieses Problem zu lösen, wird bei EED die Diffusion an Kanten nicht gänzlich eingeschränkt. Stattdessen wird lediglich Diffusion über die Kante hinweg vermieden, während Diffusion entlang der Kante erlaubt bleibt. Durch diese Bevorzugung einer Richtung wird der Diffusionsprozess anisotrop. Dieses Verhalten wird über die Konstruktion des Diffusionstensors D erreicht. Im zweidimensionalen Fall definiert sich D über zwei Eigenvektoren $\mathbf{v}_1$ und $\mathbf{v}_2$ mit zugehörigen Eigenwerten λ_1 und λ_2.

Das orthonormale System der Eigenvektoren wird so gewählt, dass der Eigenvektor $\mathbf{v}_1$ senkrecht zur Bildkante steht und $\mathbf{v}_2$ in Richtung der Kante zeigt. Um die Kantenrichtung zu ermitteln wird der Kantendetektor ∇u_σ benutzt. Dabei macht die Glättung $u_\sigma := K_\sigma * u$ den Kantendetektor robuster gegen den negativen Effekt von Bildrauschen und den Einfluss kleinster Details. Zusammenfassend ergibt sich für die Wahl der Eigenvektoren:

$$\mathbf{v}_1 \parallel \nabla u_\sigma, \quad \mathbf{v}_2 \perp \nabla u_\sigma \tag{3.8}$$

Durch die Wahl der zugehörigen Eigenwerte wird der gewünschte Effekt erzielt: λ_1 beeinflusst die Diffusion über Kanten hinweg, während λ_2 die Diffusion entlang der Kanten steuert. Daher wird λ_2 auf den Wert eins fixiert, was keiner Einschränkung der Diffusion entspricht und λ_1 wird mit einer passenden Diffusivitätsfunktion an den lokalen Kontrast $|\nabla u_\sigma|$ und damit die Kantenstärke angepasst:

$$\lambda_1(\nabla u_\sigma) := g(|\nabla u_\sigma|^2) \tag{3.9}$$

$$\lambda_2(\nabla u_\sigma) := 1 \tag{3.10}$$

Die Anpassung der Diffusion senkrecht zur Kantenrichtung wird also wie im isotropen Perona-Malik-Modell in der Nähe von Kanten anhand des Gradientenbetrages abgeschwächt.

Dreidimensionale Erweiterung

Die Erweiterung des zweidimensionalen EED-Filters auf drei Dimensionen lässt sich geradlinig aus den bereits beschriebenen Modellannahmen ableiten. Wie im zweidimensionalen Fall soll die Diffusion in Kantenrichtung begünstigt werden und über Kanten hinweg bestraft. Der Eigenvektor $\mathbf{v}_1$ des Strukturtensors $D \in \mathbb{R}^3$ wird wie im Zweidimensionalen parallel zu ∇u_σ gewählt, wobei nun jedoch $\nabla u_\sigma \in \mathbb{R}^3$ drei Komponenten enthält. Die restlichen Eigenvektoren $\mathbf{v}_2$ und $\mathbf{v}_3$ mit $\mathbf{v}_2 \perp \mathbf{v}_3 \perp \nabla u_\sigma$ spannen nun einen zweidmensionalen Unterraum des $\mathbb{R}^3$ auf. Insgesamt ergeben sich somit für die Wahl der orthonormalen Eigenvektoren die Bedingungen:

$$\mathbf{v}_1 \parallel \nabla u_\sigma \tag{3.11}$$

$$\mathbf{v}_2 \perp \nabla u_\sigma, \quad \mathbf{v}_3 \perp \nabla u_\sigma \tag{3.12}$$

$$\mathbf{v}_2 \perp \mathbf{v}_3 \tag{3.13}$$

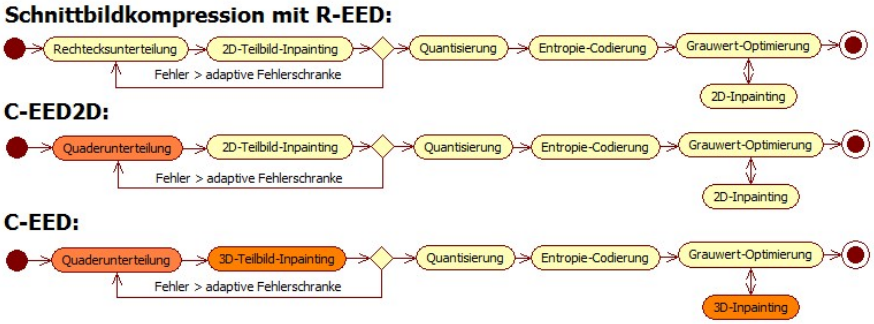


Abbildung 3.3: **UML-Übersicht: EED-Verfahren**. Die Unterschiede zwischen den verschiedenen Algorithmen R-EED, C-EED2D und C-EED sind orange hervorgehoben. R-EED und C-EED2D unterscheiden sich primär durch die Art der Bildunterteilung. Das UML-Diagramm zeigt jedoch nicht, dass R-EED auf Schnittbildern operiert und C-EED2D auf dem gesamten 3D-Bild. In C-EED2D werden die Parameter global optimiert und die Entropiecodierung profitiert von einer Grauwertliste, die das ganze Bild umfasst. C-EED erweitert C-EED2D lediglich um 3D-Inpainting.

Die zugehörigen Eigenwerte werden analog zum zweidimensionalen Fall definiert:

$$\lambda_1(\nabla u_\sigma) := g(|\nabla u_\sigma|^2) \tag{3.14}$$
$$\lambda_2(\nabla u_\sigma) = \lambda_3(\nabla u_\sigma) := 1 \tag{3.15}$$

Im kontinuierlichen Fall gestaltet sich die Erweiterung von zweidimensionaler auf dreidimensionale kantenverstärkende Diffusion als einfach. Problematischer ist die Diskretisierung, die zwar ebenfalls ähnlich wie im zweidimensionalen Fall verläuft, durch die zusätzliche Dimension jedoch merklich an Komplexität gewinnt (siehe Kapitel 4).

3.5 C-EED

Mit Hilfe der dreidimensionalen kantenverstärkenden Diffusion lässt sich nun das C-EED-Verfahren definieren. C-EED2D nutzt bereits durch den Einsatz adaptiver Quaderunterteilung Dateikopf- und Entropie-Redundanzen aus, Grauwert-Redundanzen blieben bisher allerdings ungenutzt. C-EED hebt nun die Einschränkungen der Quaderunterteilung von C-EED2D auf: die dritte Dimension wird gleichberechtigt zur x- und y-Richtung behandelt, d.h. die Bildunterteilung erfolgt immer in der größten Quaderdimension. Dadurch wird die in C-EED2D verwendete zweidimensionale Schnittbildunterteilung durch Teilbäume in eine vollwertige dreidimensionale Bildunterteilung überführt.

Der zweite Unterschied zu C-EED2D besteht im Inpaintingschritt. Die quaderförmigen Teilbilder werden mit einem dreidimensionalen EED-basierten Inpaintingverfahren rekonstruiert. Dabei ist zu beachten, dass Voxel in 3D-Raster-

bildern, die aus bildgebenden Verfahren stammen, in der Regel nicht würfelförmig sind. Die Gitterweiten sind meist von der Form $h_x = h_y = s_z h_z$ mit einem Faktor $s_z > 1$. Um optimale Inpaintingergebnisse zu gewährleisten, muss ein zusätzlicher Parameter s_z eingeführt werden, der die Gitterweite entsprechend anpasst (siehe Kapitel 4.3.2). Weitere Anpassungen des Verfahrens werden nicht benötigt.

3.6 ROI-Codierung

JPEG führt während des Kompressionsverfahrens eine Zerlegung des Bildes in quadratische Teilbilder durch. Diese Eigenschaft kann genutzt werden, um bevorzugte Bildregionen (ROI, engl. region of interest) (Strutz, 2002) auf Kosten des Restbildes bei gleichem Gesamtkompressionsverhältnis in höherer Qualität darzustellen. Von dieser Methode wird u.a. auch in der Medizin Gebrauch gemacht, um für Diagnosen wichtige Bildregionen in hoher Qualität darzustellen und das Kompressionsverhältnis zu verbessern, ohne den Rest des Bildes ganz zu verwerfen. Die mit niedrigerer Qualität codierten Regionen können auch ohne hohen Detailgrad noch dazu genutzt werden, auf einen Blick die ROI örtlich einzuordnen. Neben der offensichtlichen ROI-Codierung über die Kachelung liegen auch weitere Ergebnisse zur Kompression bevorzugter Bildregionen mit JPEG2000 vor (Bradley und Stentiford, 2002).

ROI-Codierung ist zwar nicht direkt Gegenstand dieser Arbeit, da aber medizinische Bilddaten einen signifikanten Anteil dreidimensionaler Rasterbilder ausmachen, sei hier dennoch eine natürliche Erweiterung des R-EED und Q-EED Verfahrens als Anreiz für zukünftige Forschung vorgestellt. Aufgrund der adaptiven Bildunterteilung eignen sich beide Verfahren gut für ROI-Codierung.

Im adaptiven Bildunterteilungsschritt wird über die Fehlerschranke T eine Mindestanforderung an die Rekonstruktionsqualität im aktuell betrachteten Teilbild $u_T \equiv u|_{\Omega_T}$ gestellt. Definiert man eine ROI-Maske R beliebiger Form, so kann für jedes Teilbild u_T ermittelt werden, ob die ROI-Maske u_T schneidet. Ist dies der Fall, wird eine a priori definierte adaptive Fehlerschranke $T_{\mathrm{ROI}} := a_{\mathrm{ROI}} l^d$ zur weiteren Bildunterteilung eingesetzt. Teilbilder, die keine Punkte mit der ROI-Maske teilen, nutzen die unabhängig von T_{ROI} definierte Fehlerschranke T.

Diese Entkopplung der Fehlerschranken ermöglicht durch die Wahl einer im Vergleich zu a restriktiveren Fehlerschranke a_{ROI} eine Qualitätsverbesserung innerhalb der ROI. Dabei liefert die Bildunterteilung durch neuerlichen Abgleich mit der ROI-Maske bei wachsender Baumtiefe eine genauer werdende Approximation der Maske durch Teilbildquader.

Im Optimierungsprozess der Parameter wird dabei a_{ROI} fest vorgegeben, alle anderen Parameter, auch a, werden optimiert. Dadurch wird bei gleichbleibender Kompressionsrate die Qualität außerhalb des ROI so weit verringert, wie es nötig ist, um die Qualitätssteigerung in der ROI zu erreichen. Dieser Prozess kann weiter verbessert werden, indem für eine gewünschte Kompressionsrate eine Fehlerschranke für die ROI des komprimierten Bildes vorgegeben wird. Alle Parameter inkl. a_{ROI} werden unter diesen beiden Einschränkungen optimiert.

Exemplarische Testergebnisse, die die Grundfunktionalität der ROI-Codierung mit R-EED demonstrieren, sind in Abschnitt 5.3.6 zu finden.

Kapitel 4

Numerische Aspekte und Implementierung

In den bisherigen Abschnitten wurden die theoretischen Hintergründe von R-EED, C-EED2D und C-EED behandelt und die Funktionsweise der Methoden umrissen. Um die Verfahren in der Praxis einsetzen zu können, müssen allerdings algorithmische Implementierungen aller Kompressionsschritte vorliegen.

Einige der Kompressionsschritte bleiben bei der Erweiterung von R-EED auf C-EED weitestgehend unverändert erhalten und können mit Anpassungen aus der Referenzimplementierung von Schmaltz et al. (2010) übernommen werden. Die vorgenommen Änderungen werden in Abschnitt 4.3 beschrieben.

Das 3D-Inpainting und die damit verbundene Diskretisierung der bisher nur für den kontinuierlichen Fall beschriebenen kantenverstärkenden Diffusion ist jedoch mit signifikanten Änderungen verbunden, die den Hauptteil dieses Kapitels ausmachen. Abschnitt 4.1 behandelt die Diskretisierung der EED-Diffusionsgleichung und Formulierungen des Inpaintingproblems durch lineare Gleichungssysteme (LGS). Numerische Verfahren zur Lösung des resultierenden LGS werden in Abschnitt 4.2 behandelt.

4.1 EED-Diskretisierung

In den folgenden Abschnitten wird beschrieben, wie das kontinuierliche Modell für kantenverstärkende Diffusion aus Kapitel 3.4 auf ein diskretes Bild angewendet werden kann. Hierzu ist es zunächst nötig, ein Verfahren zur Diskretisierung partieller Ableitungen zu diskutieren, da diese sowohl für die Darstellung der Diffusionsgleichung 3.3, als auch zur Berechnung des Gradientenvektorfeldes ∇u_σ benötigt werden. ∇u_σ wiederum wird eingesetzt, um den Diffusionstensor $D(\nabla u_\sigma)$ zu ermitteln.

4.1.1 Finite Differenzen

Die Diffusionsmodelle aus Kapitel 3.4 werden durch partielle Differentialgleichungen modelliert. Um diese numerisch lösen zu können, wird eine Diskretisierung partieller Ableitungen benötigt. Eine klassische Näherung von Ableitungen erfolgt durch finite Differenzen. Zunächst sei als Beispiel die Approximation der

ersten Ableitung einer Funktion $u(x) : \Omega \to \mathbb{R}$ gegeben. Die Ableitung an der Stelle $u_i := u(x)$ ist gegeben durch die *zentrale Differenz* der Nachbarpunkte $u_{i-1} := u(x-h)$ und $u_{i+1} := u(x+h)$:

$$u_i' \approx \frac{u_{i+1} - u_{i-1}}{2h} \tag{4.1}$$

Unter der Voraussetzung, dass u auf Ω $n+1$ mal stetig differenzierbar ist, lässt sich u in jedem Punkt durch eine Taylorreihe approximieren. Im eindimensionalen Fall hat diese die Form:

$$u(x+h) = \sum_{k=0}^{n} \frac{h^k}{k!} f^{(k)}(x) + \mathcal{O}(h^{n+1}) \tag{4.2}$$

$$\text{mit } f \in \mathcal{O}(g) :\Leftrightarrow 0 \leq \limsup_{x \to a} \frac{|f(x)|}{|g(x)|} < \infty, \quad a \in \mathbb{R} \cup \{-\infty, \infty\} \tag{4.3}$$

$\mathcal{O}$ bezeichnet hierbei das in Gleichung 4.3 definierte *Landau-Symbol*, welches zur Einordnung des asymptotischen Verhaltens von Funktionen dient. $f \in \mathcal{O}$ bedeutet, dass die Funktion f bis auf einen skalaren Vorfaktor betragsmäßig nicht schneller wächst als die Funktion g. Da im konkreten Fall der Fehlerapproximation bei finiten Differenzen $h \leq 1$ gilt, ist der Term mit der niedrigsten Potenz von h als größter Fehlerterm ausschlaggebend.

Mit Hilfe der Gleichung 4.2 können nun Taylorapproximationen von u_{i+1} und u_{i-1} berechnet werden:

$$u(x-h) = u_{i-1} = u_i - hu_i' + \frac{h^2}{2}u_i'' - \frac{h^3}{6}u_i''' + \frac{h^4}{24}u_i'''' + \mathcal{O}(h^5) \tag{4.4}$$

$$u(x+h) = u_{i+1} = u_i + hu_i' + \frac{h^2}{2}u_i'' + \frac{h^3}{6}u_i''' + \frac{h^4}{24}u_i'''' + \mathcal{O}(h^5) \tag{4.5}$$

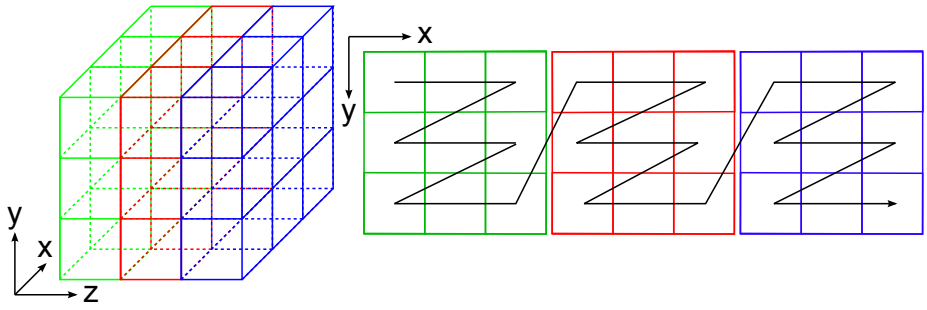
Durch Einsetzen der Approximationen 4.4 und 4.5 in die zentrale Differenz 4.1 lässt sich deren führender Fehlerterm berechnen:

$$\begin{aligned} \frac{u_{i+1} - u_{i-1}}{2h} &= \underbrace{\frac{1}{2h}(1-1)}_{=0} u + \underbrace{\frac{1}{2}(1+1)}_{=1} u_i' + \underbrace{\frac{h}{4}(1-1)}_{=0} u_i'' + \underbrace{\frac{h^2}{12}(1+1)}_{\neq 0} u_i''' + \mathcal{O}(h^3) \\ &= u_i' + \mathcal{O}(h^2) \end{aligned}$$

Der Exponent der Gitterweite h im führenden Fehlerterm definiert hierbei die *Konsistenzordnung* der Approximation. Die Näherung der ersten Ableitung 4.1 hat also Konsistenzordnung 2.

Partielle Ableitungen in höherdimensionalen Fällen können auf gleiche Weise durch finite Differenzen modelliert werden. Die Approximation erfolgt dabei durch mehrdimensionale Taylorreihen. Für die erste partielle Ableitung nach einer Variablen x bleiben jedoch nur die partiellen Ableitungsterme nach x in der Taylorreihe erhalten, d.h. die Approximation 4.4 und 4.5 lassen sich analog auf partielle Ableitungen höherer Dimensionen übertragen.

Auf diese Weise lassen sich für den in dieser Arbeit betrachteten dreidimen-



(a)

Abbildung 4.1: **3D-Daten in Vektordarstellung** Die Abbildung demonstriert, wie in Raumkoordinaten gegebene 3D-Daten in eine Vektordarstellung überführt werden. Die endliche Menge von Punkten wird beginnend mit dem hinteren (negative z-Richtung) linken (negative x-Richtung) oberen (negative y-Richtung) Eckpunkt durchnumeriert. Dabei erfolgt die Numerierung pro Zeile von links nach rechts für alle Zeilen eines jeden x-y-Schnittbildes in positiver y-Richtung. Die Schnittbilder werden in positiver z-Richtung durchlaufen.

sionalen Fall die folgenden finiten Differenzen definieren:

$$\partial_x u_{i,j,k}, = \frac{u_{i+1,j,k} - u_{i-1,j,k}}{2h_x} + \mathcal{O}(h_x^2) \quad \text{(Zentrale Differenz)} \tag{4.6}$$

$$\partial_x u_{i,j,k} = \frac{u_{i+1,j,k} - u_{i,j,k}}{h_x} + \mathcal{O}(h_x) \quad \text{(Vorwärtsdifferenz)} \tag{4.7}$$

$$\partial_x u_{i,j,k} = \frac{u_{i,j,k} - u_{i-1,j,k}}{h_x} + \mathcal{O}(h_x) \quad \text{(Rückwärtsdifferenz)} \tag{4.8}$$

Um den Diffusionstensor in Abschnitt 4.1.2 zu berechnen, werden auch Ableitungen an Zwischenstellen in halber Gitterweite benötigt. Dabei lässt sich ausnutzen, dass Vorwärts- und Rückwärtsdifferenzen die Ableitung in Zwischenstellen wie eine zentrale Differenz mit Konsistenzordnung 2 approximieren:

$$\partial_x u_{i+\frac{1}{2},j,k} = \frac{u_{i+1,j,k} - u_{i,j,k}}{h_x} + \mathcal{O}(h_x^2) \tag{4.9}$$

4.1.2 Semi-Diskretisierung

Es gilt, die folgende Diffusionsgleichung 4.10 mit Diffusionstensor D mit Hilfe finiter Differenzen in eine diskrete Darstellung zu überführen:

$$\partial_t u = \operatorname{div}(D \nabla u) \tag{4.10}$$

$$D = \begin{pmatrix} a & b & c \\ b & d & e \\ c & e & f \end{pmatrix} \tag{4.11}$$

Das Ziel ist hierbei zunächst, die Zeitableitung jedes Bildpunkts $\partial_t u_{i,j,k}$ durch eine gewichtete Summe seiner $3 \times 3 \times 3$-Nachbarschaft aus dem vorangehenden Zeitschritt t zu berechnen. Da die künstliche Zeitdimension zunächst

nicht diskretisiert wird, wird dieser erste Diskretisierungsschritt auch als *Semi-Diskretisierung* bezeichnet. Die Gewichte ergeben sich dabei aus Einträgen des Diffusionstensors. Hierbei ist zu beachten, dass EED ein nichtlinearer Diffusionsprozess ist. Daher ändert sich der Wert des Diffusionstensors abhängig von der betrachteten Position im Bild. Die Einträge des Diffusionstensors $D_{i,j,k}$ werden somit ebenfalls mit Raumkoordinaten versehen.

Um die Komplexität der Schreibweise zu reduzieren, wird im Folgenden eine Vektorschreibweise für diskrete 3D-Daten benutzt. Die Bildpunkte werden anhand der Reihenfolge aus Abbildung 4.1 mit einem einzelnen Index versehen, der die Index-Tripel der Raumdarstellung ersetzt. Der Zusammenhang zwischen beiden Indizes ist für einen 3D-Datensatz v mit Dimension $n_x \times n_y \times n_z$ durch die folgende Gleichung beschrieben:

$$\begin{aligned} \mathbf{v}_\alpha = v_{i,j,k} \quad &\text{mit} \quad k := \lfloor \alpha / n_z \rfloor + 1, j := \lfloor (\alpha - k n_z)/n_y \rfloor + 1, i := \alpha - j n_y - k n_z \\ &\text{bzw.} \quad \alpha := (k-1) n_z + (j-1) n_y + i \end{aligned}$$

Zusammen mit der Vektordarstellung wird zudem eine neue Mengennotation eingeführt: $(i,j,k) \in \Omega \Leftrightarrow \alpha \in \Omega_V$. In selbiger Vektordarstellungen sei im Folgenden die Menge $\mathcal{N}(u_{i,j,k}) = \{x_1, \ldots, x_n\}$ der $3 \times 3 \times 3$ Nachbaren des Punktes $u_{i,j,k}$ bezeichnet. Die Gewichte für diese Nachbarschaft erhalten die Bezeichnung $\mathcal{W}(u_{i,j,k}) = \{w_1, \ldots, w_n\}$ und bilden eine sogenannten *Schablone* (engl. *stencil*). Die Anwendung der Schablone auf den Bildpunkt $u_{i,j,k}$ liefert die Approximation der Zeitableitung:

$$\partial_t u_{i,j,k} \approx \sum_{l=1}^{n} w_l x_l, \quad x_l \in \mathcal{N}(u_{i,j,k}), w_l \in \mathcal{W}(u_{i,j,k}). \tag{4.12}$$

In der Regel werden die räumlichen Ableitungen aus $\operatorname{div}(D\nabla u)$ und die Zeitableitung $\partial_t u$ in einer *Standarddiskretisierung* durch zentrale Differenzen approximiert. Da die Randbedingungen $\langle D\nabla u, \mathbf{n} \rangle = 0$ aus Gleichung 3.6 für diesen Fall nicht direkt ersichtlich sind, wird in dieser Arbeit eine andere Form der Standarddiskretisierung, basierend auf bisher unveröffentlichten Ergebnissen von Weickert (2010) gewählt. Es ergeben sich die folgenden 27 Schablonen-Gewichte:

$$\begin{aligned}
w_1 &= w_3 = w_7 = w_9 = w_{19} = w_{21} = w_{25} = w_{27} = 0 \\
w_2 &= (e_{i-\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}} + e_{i+\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}})(8h_y h_z)^{-1} \\
w_4 &= (c_{i-\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}} + c_{i-\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}})(8h_x h_z)^{-1} \\
w_5 &= (f_{i+\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}} + f_{i+\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}} + f_{i-\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}} + f_{i-\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}})(16h_z^2)^{-1} \\
w_6 &= -(c_{i+\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}} + c_{i+\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}})(8h_x h_z)^{-1} \\
w_8 &= -(e_{i-\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}} + e_{i+\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}})(8h_y h_z)^{-1} \\
w_{10} &= (b_{i-\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}} + b_{i-\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}})(8h_x h_y)^{-1} \\
w_{11} &= (d_{i-\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}} + d_{i-\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}} + d_{i+\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}} + d_{i+\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}})(16h_y^2)^{-1} \\
w_{12} &= -(b_{i+\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}} + b_{i+\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}})(8h_x h_y)^{-1} \\
w_{13} &= (a_{i-\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}} + a_{i-\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}} + a_{i-\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}} + a_{i-\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}})(16h_x^2)^{-1} \\
w_{14} &= -\sum_{i \neq 14} w_i
\end{aligned}$$

$$
\begin{aligned}
w_{15} &= (a_{i+\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}} + a_{i+\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}} + a_{i+\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}} + a_{i+\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}})(16h_x^2)^{-1} \\
w_{16} &= -(b_{i-\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}} + b_{i-\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}})(8h_xh_y)^{-1} \\
w_{17} &= (d_{i+\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}} + d_{i-\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}} + d_{i+\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}} + d_{i-\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}})(16h_y^2)^{-1} \\
w_{18} &= (b_{i+\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}} + b_{i+\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}})(8h_xh_y)^{-1} \\
w_{20} &= -(e_{i-\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}} + e_{i+\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}})(8h_yh_z)^{-1} \\
w_{22} &= -(c_{i-\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}} + c_{i-\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}})(8h_xh_z)^{-1} \\
w_{23} &= (f_{i+\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}} + f_{i-\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}} + f_{i+\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}} + f_{i-\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}})(16h_z^2)^{-1} \\
w_{24} &= (c_{i+\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}} + c_{i+\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}})(8h_xh_z)^{-1} \\
w_{26} &= (e_{i-\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}} + e_{i+\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}})(8h_yh_z)^{-1}
\end{aligned}
$$

Randbedingungen lassen sich über Dirichlet-Bedingungen für die Bildpunkte $u_{i,j,k}$ und die Einträge des Strukturtensors $D_{i,j,k}$ definieren. Setzt man $u_{i,j,k} = 0$ und $D_{i,j,k} = 0$ (Nullmatrix) für alle $(i,j,k)^T \notin \Omega$, so gilt Gleichung 4.12 mit den oben definierten Gewichten auch auf dem Rand $\partial\Omega$.

Während sich die Darstellung der Diskretisierung der EED-Diffusionsgleichung durch eine Schablone gut für die konkrete Implementierung eignet, ist für die Ableitung von Lösungsverfahren eine Formulierung durch eine Matrix-Vektor-Multiplikation $\partial_t u \approx A\mathbf{u}$ üblich. Jede Anwendung einer Schablone auf einen Punkt $u_{i,j,k}$ gemäß Gleichung 4.12 definiert eine Zeile der Matrix, die bis auf die Schablonengewichte nur Nullen enthält. Die Grauwerte des Bildes $u_{i,j,k}$ werden in die Vektorform aus Abbildung 4.1 gebracht. Daraus ergeben sich die Dimensionen $N = n_x \cdot n_y \cdot n_z$ für $\mathbf{u} \in \mathbb{R}^N$, sowie $A \in \mathbb{R}^{N \times N}$

4.1.3 Diskretisierungsansätze

Ist eine Semi-Diskretisierung einer Diffusionsgleichung gegeben, so bieten sich verschiedene Möglichkeiten an, das Problem vollständig, also auch in der Zeitdimension, zu diskretisieren und damit einen Ansatz für ein Lösungsverfahren zu definieren. Dazu wird die Zeitdimension durch eine finite Differenz mit Zeitschrittweite τ approximiert.

Auf diese Weise lässt sich ein *explizites* Verfahren herleiten, welches direkt erlaubt, das Ergebnis u^{k+1} des nächsten Zeitschritts aus dem Bild u^k des vorherigen Zeitschritts zu berechnen:

$$\frac{\mathbf{u}^{k+1} - \mathbf{u}^k}{\tau} = A(\mathbf{u}^k)\mathbf{u}^k \tag{4.13}$$

$$\Leftrightarrow \quad \mathbf{u}^{k+1} = (I + \tau A(\mathbf{u}^k))\mathbf{u}^k \tag{4.14}$$

Explizite Verfahren haben jedoch den Nachteil, dass sie nur für kleine Zeitschrittweiten τ Stabilität garantieren (Weickert, 1996a). Zwar liegen neue, vielversprechende Ansätze vor (Grewenig et al., 2010, siehe auch Kapitel 6.2), welche die Nachteile expliziter Verfahren ausgleichen können, im Rahmen dieser Arbeit wird jedoch stattdessen ein *semi-implizites* Verfahren eingesetzt. Der Ansatz unterscheidet sich dabei auf den ersten Blick nur leicht von dem des expliziten

Schemas:

$$\frac{\mathbf{u}^{k+1}-\mathbf{u}^k}{\tau} = A(\mathbf{u}^k)\mathbf{u}^{k+1} \tag{4.15}$$

$$\Leftrightarrow \quad \mathbf{u}^{k+1} = (I - \tau A(\mathbf{u}^k))^{-1}\mathbf{u}^k \tag{4.16}$$

Semi-implizite Verfahren sind zwar stabiler als explizite Verfahren (Weickert, 1996a), allerdings kann hier nur der Diffusionstensor und damit die Matrix $A(\mathbf{u}^k)$ direkt berechnet werden. Um die Grauwerte des Bildes $\mathbf{u}^{k+1}$ zu ermitteln, muss ein lineares Gleichungssystem mit der Systemmatrix $I-\tau A(\mathbf{u}^k)$ gelöst werden. Methoden zur Lösung linearer Gleichungssysteme werden in Abschnitt 4.2 beschrieben.

Das Gleichungssystem $(I - \tau A(\mathbf{u}^k))\mathbf{u}^{k+1} = \mathbf{u}^k$ beschreibt lediglich den anisotropen Diffusionsprozess, nicht das benötigte PDE-basierte Inpainting. Nach Kapitel 2.1 genügt jedoch eine leichte Modifikation der Matrix $A(\mathbf{u}^k)$, um die Interpolations-PDE zu diskretisieren. Statt dass die fixierten Grauwerte der Interpolationsmaske B direkt in den Diffusionsprozess einbezogen werden, definieren sie Dirichlet-Randbedingungen $\mathbf{u}_i^k = \mathbf{f}_i$ für B. Zusammen mit den bestehenden Bedingungen für $\partial\Omega$ ergeben sich die neuen Randbedingungen des Verfahrens.

Für die Bildpunkte in B können diese Randbedingungen entweder dadurch dargestellt werden, dass für jeden Punkt $i \in B_V$ das zentrale Schablonengewicht auf eins und die restlichen Gewichte auf null gesetzt werden. Dadurch entsteht eine Matrix $\tilde{A}(\mathbf{u}^k)$, deren i-te Zeile der von $A(\mathbf{u}^k)$ entspricht, wenn $i \in \Omega_V \setminus B_V$ und andernfalls die Form des i-ten Einheitsvektors e_i annimmt. Mainberger und Weickert (2009) bezeichnen diese Form als *erweiterte* (engl. *extended*) Systemmatrix. Das zugehörige Gleichungssystem für die explizite Darstellung ist von der Form:

$$\mathbf{u}_i^{k+1} = (1 - \chi_B(\mathbf{u}_i^k)) \sum_{l=1}^{n} w_l \mathbf{u}_l^k + \chi_B \mathbf{u}_i^k, \quad \mathbf{u}_l^k \in \mathcal{N}(\mathbf{u}_i^{k+1}), w_l \in \mathcal{W}(\mathbf{u}_i^k). \tag{4.17}$$

Demgegenüber steht die *reduzierte* Darstellung, in der lediglich die Gleichungen in $\mathbf{u}_i^{k+1}$, $i \in \Omega_V \setminus B_V$ betrachtet werden und die Dirichlet-Randbedingungen angewendet werden. Da beide Darstellungen das gleiche Ergebnis liefern, sind sie äquivalent.

4.1.4 Eigenschaften der Diskretisierung

Der Schwerpunkt dieser Arbeit liegt auf der praktischen Anwendung der in diesem Kapitel beschriebenen Diskretisierung, weshalb theoretische Aspekte wie Gutgestelltheit und Skalenraumeigenschaften nicht untersucht werden. Für die effiziente Implementierung der Diskretisierung in einer Programmiersprache sind jedoch einige der Eigenschaften von $A(\mathbf{u}^k)$ von Bedeutung: Die Schablonengewichte $\mathcal{W}(u_{i,j,k})$ lassen sich auch durch die Schablonenkomponenten benachbarter Bildpunkte darstellen. Dies sei am Beispiel der Gewichte $w_8^{i,j,k} \in \mathcal{W}(u_{i,j,k})$ und $w_{20}^{i,j+1,k-1} \in \mathcal{W}(u_{i,j+1,k-1})$ erläutert:

$$\begin{aligned} w_8^{i,j,k} &= -(e_{i-\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}} + e_{i+\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}})(8h_yh_z)^{-1} \\ &= -(e_{i-\frac{1}{2},(j+1)-\frac{1}{2},(k-1)+\frac{1}{2}} + e_{i+\frac{1}{2},(j+1)-\frac{1}{2},(k-1)+\frac{1}{2}})(8h_yh_z)^{-1} \\ &= w_{20}^{i,j+1,k-1} \end{aligned}$$

Daher muss nur die Hälfte der nichtzentralen Schablonengewichte berechnet werden, der Rest ergibt sich aus den Schablonen der Nachbarpunkte. Da sich das mittlere Schablonengewicht w_{14} als Summe der restlichen Gewichte berechnen lässt und acht der Gewichte immer den Wert Null haben, müssen insgesamt nur neun Gewichte pro Bildpunkt explizit berechnet werden.

4.2 SOR-Verfahren

SOR (engl. *successive-overrelaxation*) ist ein iteratives Verfahren zur Lösung von linearen Gleichungssystemen der Form $A\mathbf{x} = \mathbf{b}$ mit $A \in \mathbb{R}^{n\times n}$ und $\mathbf{x}$, $\mathbf{b} \in \mathbb{R}^n$. Stationäre iterative Verfahren wie SOR beruhen auf dem Grundprinzip, ausgehend von einem Anfangswert $\mathbf{x}^{(0)}$ sukzessiv eine rekursiv definierte Folge $(\mathbf{x}^{(k)})_{k\in\mathbb{N}}$ zu berechnen, die gegen die Lösung $\mathbf{x}$ konvergiert. Allgemein lassen sich stationäre Iterationsverfahren über die rekursive Definition von $\mathbf{x}^{(k)}$ beschreiben:

$$\mathbf{x}^{(k)} = B\mathbf{x}^{(k-1)} + \mathbf{c} \quad B \in \mathbb{R}^{n\times n}, \mathbf{c} \in \mathbb{R}^n \tag{4.18}$$

Die folgenden Abschnitte geben einen Überblick über die Funktionsweise von SOR und die Herleitung des SOR-Verfahrens über die Jacobi- und Gauß-Seidel-Methoden. Diese Übersicht orientiert sich lose an Kapitel 2 von "Templates for the Solution of Linear Systems"(Barrett et al., 1994), wo auch weitere Informationen zum Konvergenzverhalten dieser Verfahren zu finden sind.

4.2.1 Jacobi-Verfahren

Im Jacobi-Verfahren werden die Gleichungen des Systems $A\mathbf{x} = \mathbf{b}$ getrennt betrachtet. Für jede Gleichung $(A\mathbf{x})_i = \mathbf{b}_i$ werden die Werte $\mathbf{x}_j$ mit $j \neq i$ als Konstanten betrachtet und die Gleichung nach $\mathbf{x}_i$ gelöst:

$$\sum_{j=1}^{n} a_{i,j}\mathbf{x}_j = \mathbf{b}_i \tag{4.19}$$

$$\Leftrightarrow \quad \mathbf{x}_i = \frac{\left(\mathbf{b}_i - \sum_{j\neq i} a_{i,j}\mathbf{x}_i\right)}{a_{i,i}} \tag{4.20}$$

Das von Gleichung 4.20 abgeleitete Iterationsverfahren lässt sich somit direkt in der Form 4.18 darstellen:

$$\mathbf{x}_i^{(k)} = \underbrace{\frac{\mathbf{b}_i}{a_{i,i}}}_{=:c_k^J} \underbrace{- \sum_{j\neq i} \frac{a_{i,j}}{a_{i,i}} \mathbf{x}_j^{(k-1)}}_{=:B_k^J} \tag{4.21}$$

Auch wenn der Fehler $\mathbf{x} - \mathbf{x}^{(k)}$ nicht aus der Unbekannten $\mathbf{x}$ errechnet werden kann, so lässt sich jedoch die Auswirkung dieses Fehlers auf die Lösung des Gleichungssystems, im Weiteren als *Residuum* $\mathbf{r}^{(k)}$ bezeichnet (Meister, 1999), betrachten:

$$\mathbf{r}^k := A(\mathbf{x} - \mathbf{x}^k) = A\mathbf{x} - A\mathbf{x}^k = \mathbf{b} - A\mathbf{x}^k \tag{4.22}$$

Eine genauere Betrachtung der Gleichung 4.21 zeigt, dass sich das Jacobi-Verfahren auch über das Residuum beschreiben lässt:

$$\mathbf{x}_i^{k+1} = \frac{\mathbf{b}_i}{a_{i,i}} - \sum_{j \neq i} \frac{a_{i,j}}{a_{i,i}} \mathbf{x}_j^{k-1} = \frac{\mathbf{b}_i}{a_{i,i}} + \frac{a_{i,i}\mathbf{x}_i^k - (A\mathbf{x}^k)_i}{a_{i,i}} \tag{4.23}$$

$$= \mathbf{x}_i^k + \frac{(\mathbf{b} - A\mathbf{x}^k)_i}{a_{i,i}} = \mathbf{x}_i^k - \underbrace{\frac{\mathbf{r}_i^k}{a_{i,i}}}_{\text{Korrekturfaktor}} \tag{4.24}$$

Diese Darstellung lässt die Interpretation des Jacobi-Verfahrens als Abfolge von Korrekturschritten zu, wobei sich der jeweilige Korrekturfaktor aus dem passend skalierten Residuum ergibt.

4.2.2 Gauß-Seidel-Verfahren

Das Gauß-Seidel-Verfahren kann als eine Erweiterung des Jacobi-Verfahrens interpretiert werden. Im Jacobi-Verfahren werden die n Gleichungen 4.20 in jedem Schritt völlig unabhängig voneinander betrachtet. Berechnet man die Werte $\mathbf{x}_1^{(k)}, \ldots, \mathbf{x}_n^{(k)}$ sequentiell, so liegen zur Zeit der Berechnung von $\mathbf{x}_i^{(k)}$ schon die Ergebnisse für $\mathbf{x}_j^{(k)}$ mit $j < i$ vor. Verwendet man diese in Gleichung 4.21, so erhält man:

$$\mathbf{x}_i^{(k)} = \frac{\mathbf{b}_i}{a_{i,i}} - \left(\sum_{j<i} \frac{a_{i,j}}{a_{i,i}} \mathbf{x}_j^{(k)} + \sum_{j>i} \frac{a_{i,j}}{a_{i,i}} \mathbf{x}_j^{(k-1)} \right) \tag{4.25}$$

Durch die Verwendung dieser Zwischenergebnisse wird im Allgemeinen eine schnellere Konvergenz zum Lösungsvektor erreicht. Dieser Vorteil wird allerdings durch die Abhängigkeit des Ergebnisses des Folgeschrittes von der Reihenfolge der Gleichungen erkauft. Der Korrekturfaktor ist dabei bis auf die Verwendung der Zwischenergebnisse derselbe wie im Jacobi-Verfahren.

4.2.3 Sukzessive Überrelaxation

Die SOR Methode beruht auf dem gewichteten Mittel zwischen der Iterierten $\mathbf{x}_i^{(k)}$ des Gauß-Seidel-Verfahrens und $\mathbf{x}_i^{(k-1)}$.

$$\mathbf{x}_i^{(k)} = \omega \left(\frac{\mathbf{b}_i}{a_{i,i}} - \left(\sum_{j<i} \frac{a_{i,j}}{a_{i,i}} \mathbf{x}_j^{(k)} + \sum_{j>i} \frac{a_{i,j}}{a_{i,i}} \mathbf{x}_j^{(k-1)} \right) \right) + (1-\omega)\mathbf{x}_i^{(k-1)} \tag{4.26}$$

Für $\omega = 1$ verhält sich das SOR-Verfahren identisch zur Gauß-Seidel-Methode. Für eine andere Wahl von ω wird der Korrekturfaktor der Gauß-Seidel-Methode entweder gedämpft ($\omega < 1$) oder begünstigt ($\omega > 1$, *Überrelaxation*). Durch eine Überrelaxation kann die Konvergenzgeschwindigkeit erhöht werden.

Die Wahl von ω bestimmt nicht nur, ob SOR einen Gewinn gegenüber Gauß-Seidel in der Konvergenzgeschwindigkeit erreicht, sondern beeinflusst auch das Konvergenzverhalten. Unabhängig von der Beschaffenheit der Koeffizientenmatrix A konvergiert SOR nur für $\omega \in (0, 2)$ (Varga, 1962). Für symmetrische, positiv definite Matrizen ist eine Konvergenz für alle Werte von $\omega \in (0, 2)$

nachgewiesen. Einen optimalen Parameter ω zu berechnen, der die schnellstmögliche Konvergenz garantiert, ist theoretisch über den Spektralradius der Jacobi-Iterationsmatrix gegeben, wird aber in der Praxis wegen des hohen Rechenaufwandes nicht eingesetzt.

In Bezug auf die konkrete Anwendung von SOR im C-EED-Verfahren sei angemerkt, dass SOR aufgrund der Abhängigkeit des Gauß-Seidel-Verfahrens von der Gleichungsreihenfolge verschiedene negative Eigenschaften aufweist. Zum einen wird SOR direkt von der Wahl der Vektordarstellung des Bildes (vgl. Abb. 4.1) beeinflusst, zum anderen führt dies auch zu einem Verlust der Rotationsinvarianz des Jacobi-Verfahrens. Eine einfache Rotation des Bildes in 90°-Schritten genügt bereits, um das Verhalten des SOR-Verfahrens zu verändern.

4.2.4 Sukzessive Verfeinerung

SOR ist ein iteratives Verfahren, welches ausgehend von einem Initialwert $\mathbf{x}^0$ durch sukzessive Anwendung derselben algorithmischen Schritte eine Folge $\mathbf{x}^k$ berechnet, die gegen die gewünschte Lösung $\mathbf{x}$ konvergiert. Die Wahl von $\mathbf{x}^0$ beeinflusst dabei die Anzahl der notwendigen Iterationen: je mehr $\mathbf{x}^0$ bereits der Lösung entspricht, desto weniger Iterationen werden benötigt.

Diesen Umstand machen sich *sukzessive Verfeinerungsverfahren* zu Nutze. Das Eingabebild $\mathbf{u}$, welches auch als Initalwert fungiert, wird dabei zunächst in mehreren Stufen rekursiv auf eine niedrigere Auflösung eingeschränkt. Es entsteht eine Bildpyramide, deren Auflösung mit steigender Tiefe jeweils halbiert wird. Auf der tiefsten Rekursionsstufe wird das iterative Verfahren auf das Bild mit der niedrigsten Auflösung angewendet. Das Ergebnis dieser Iteration wird dann auf die doppelte Auflösung interpoliert und im rekursiven Aufstieg an die nächst höhere Rekursionsstufe als Initialwert für das Iterationsverfahren weitergegeben. Da für niedrige Auflösungen die Lösungsverfahren weniger Laufzeit benötigen und das interpolierte Ergebnis der Lösung auf höheren Auflösungen näher kommt, wird damit die Konvergenzgeschwindigkeit auf den laufzeitintensiven, hohen Auflösungen verbessert.

Das Prinzip der sukzessiven Verfeinerung wird beispielsweise zur Schätzung von Bewegungsinformationen in Bildsequenzen eingesetzt (Black und Anandan, 1996; Mémin und Pérez, 2002) und ist im weitesten Sinne mit mathematischen Mehrgitterverfahren (Briggs, 1987) verwandt.

4.3 ANSI-C Implementierung

Die Referenzimplementierung der C-EED und C-EED2D nutzt die Programmiersprache ANSI-C und baut direkt auf die R-EED-Implementierung von Schmaltz et al. (2010) auf. Im Folgenden wird ein Überblick über die Unterschiede zwischen R-EED und C-EED gegeben. Eine Kurzübersicht über die Bedienungsmöglichkeiten der Referenzimplementierung ist in Anhang B zu finden.

4.3.1 Datenstrukturen und Dateiformat

Die R-EED-Implementierung besteht im Wesentlichen aus dem Quaderunterteilungsalgorithmus, der über eine Filterschnittstelle mit verschiedenen Inpainting-

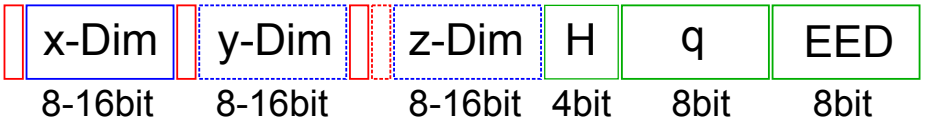


Abbildung 4.2: **Dateikopf für C-EED-Dateien** Der Dateikopf enthält die Bilddimensionen (blau), Kompressionsparameter (grün) und die im Bild nicht dargestellte Codierung des Binärbaums aus dem Bildunterteilungsschritt. Die rot markierten Bits sind 1-bit-Schalter, die eine effiziente Speicherung der Bilddimensionen erlauben. Der erste Schalter bestimmt, ob 8 bit für die Darstellung der Bilddimensionen ausreichen, oder 16 bit benötigt werden. Die Schalter zwei bis vier werden genutzt, wenn mehrere Bilddimensionen identisch sind, um keine davon doppelt speichern zu müssen.

Modulen und über eine weitere Schnittstelle mit Entropie-Verfahren Daten austauschen kann. Das daraus resultierende Kompressionsverfahren wird zudem durch Algorithmen erweitert, die die Optimierung verschiedener Parameterkombinationen erlauben.

Diese Grundstruktur wurde beibehalten, allerdings sind nahezu alle Programmteile von Änderungen in den Datenstrukturen und Funktionsparametern betroffen. Die Erweiterung um eine zusätzliche Dimension erfordert eine Änderung sämtlicher Interaktionen des Programms mit den Grauwerten, da diese in einem dreidimensionalen Array gespeichert werden. Lediglich die Algorithmen zur Entropie-Codierung sowie die Parameteroptimierung sind nicht von diesen Änderungen betroffen: Entropie-Codierungen behandeln die Grauwerte sequentiell und unabhängig von ihrer Position, die Schnittstelle kann also weiterhin mit einer eindimensionalen Grauwertliste operieren. Die Parameteroptimierung ist weitestgehend unabhängig von der Ausprägung der Bilddatenstrukturen, da der Kompressionsalgorithmus nur aufgerufen und die resultierenden Ergebnisse betrachtet werden. Lediglich die Fehlerberechnung muss angepasst werden.

Der Aufbau des Dateikopfes der C-EED-Ausgabedateien ist in Abbildung 4.2 dargestellt. Die einzige Änderung gegenüber R-EED betrifft die effiziente Speicherung der dritten Dimension. Dazu werden neue 1-bit-Schalter eingeführt, welche die Information enthalten ob zwei oder drei Bilddimensionen identisch sind, um so Speicherplatz für redundante Dimensionen einzusparen.

4.3.2 Inpainting

Die Filterschnittstelle ist in C-EED an die Erfordernisse der 3D-Daten angepasst, behält aber ihre Funktionalität. Drei verschiedene, neue Inpainting-Module sind in der Referenzimplementierung enthalten: ein explizites Lösungsverfahren basierend auf der 3D-EED-Standarddiskretisierung, sowie je eine SOR-Implementierung mit sukzessiver Verfeinerung für 2D-EED und 3D-EED.

Das explizite Lösungsverfahren stellt ein vollständig neues Inpainting-Modul dar, während die zweidimensionale SOR-Implementierung aus dem R-EED-Verfahren übernommen und auf die neue Inpainting-Schnittstelle angepasst wurde. 3D-Daten werden von dem zweidimensionalen Verfahren als Sequenz von Schnittbildern in x-y-Richtung interpretiert, die nacheinander abgearbeitet werden.

Das dreidimensionale SOR-Verfahren basiert zwar auf der im R-EED-Verfahren eingesetzten Implementierung, hat jedoch einschneidende strukturelle und funktionale Änderungen erhalten. Die Methoden zur Speicherallokation der verwendeten dreidimensionalen Bildarrays und vierdimensionalen Bildpyramiden für die sukzessive Verfeinerung wurden komplett neu implementiert und als eigenständiges Modul aus dem Verfahren ausgekoppelt.

Ähnliches gilt für die im SOR-Verfahren verwendeten Routinen für Gauß-Glättung und Bild-Restriktion/Interpolation, die jeweils für die Berechnung des Diffusionstensors bzw. zur Erstellung der Bildpyramide benötigt werden. Beide auf drei Dimensionen erweiterte Algorithmen werden in der C-EED-Implementierung ebenfalls in eigenständigen Modulen zur Verfügung gestellt.

Weiterhin wird die Diskretisierung der EED-PDE durch die in Kapitel 4.1.2 beschriebene ersetzt und die Berechnung des Diffusionstensors entsprechend angepasst. Dabei werden die im selben Abschnitt beschriebenen Symmetrien ausgenutzt um Rechenzeit und Speicherplatzbedarf zu reduzieren. Das SOR-Verfahren selbst bleibt bis auf die Anpassung der Datenstrukturen unberührt, da es implizit auf einer Vektordarstellung des Bildes operiert, die Dimensionen des Ursprungsbildes dabei also keine Rolle spielen.

Gitterweiten

Ein wichtiger Unterschied zum zweidimensionalen Fall ergibt sich bei der Anwendung von EED in drei Dimensionen durch die technisch bedingte Ungleichbehandlung der dritten Dimension. Bei 2D-Rasterbildern haben die Pixel für gewöhnlich eine quadratische Form, d.h. $h_x = h_y$. Bildgebende Verfahren nutzen aus technischen Gründen für die dritte Dimension jedoch oft eine niedrigere Abtastrate als für die anderen beiden Dimensionen.

Diese Unterschiede in den Gitterweiten müssen im Inpaintingschritt berücksichtigt werden, weshalb ein neuer Parameter $s_z > 0$ eingeführt wurde. Im Gegensatz zur R-EED-Implementierung, in der alle Gitterweiten auf den Wert eins fixiert werden, kann bei C-EED eine Anpassung durch den Parameter s_z vorgenommen werden. Die Anpassung beruht auf den beiden Grundannahmen:

$$h_x = h_y = s_z h_z \tag{4.27}$$

$$h_z = 1 \tag{4.28}$$

Die Gitterweiten h_x und h_y werden daher durch den Faktor s_z auf einen passenden Wert $h_x = h_y = \frac{h_z}{s_z} = \frac{1}{s_z}$ skaliert. Der Standardwert für s_z ist dabei 1, ohne Parameterübergabe werden also würfelförmige Voxel mit Gitterweiten $h_x = h_y = h_z = 1$ angenommen.

4.3.3 ROI-Codierung

Die ROI-Codierung ist eine neue Funktionalität für C-EED, die jedoch in gleicher Weise auch in das R-EED-Verfahren integriert werden kann. Die Implementierung orientiert sich dabei eng an der Beschreibung des Verfahrens aus Kapitel 3.6. Da die ROI-Codierung nur ein Nebenprodukt dieser Arbeit darstellt und lediglich rudimentärer Natur ist, ist diese Funktionalität in der Referenzimplementierung durch Compilerweichen standardmäßig deaktiviert.

Kapitel 5

Experimente und Analyse

5.1 Ziele

Im folgenden Kapitel werden Experimente beschrieben und analysiert, die verschiedene Eigenschaften des C-EED-Algorithmus untersuchen. Hauptziel ist dabei der Vergleich mit R-EED und die Einschätzung des Qualitätsgewinns durch die Ausnutzung der in Kapitel 3.1 beschriebenen Redundanzen und Irrelevanzen im dreidimensionalen Bild. Hauptziel der Experimente ist also die Beantwortung der folgenden Fragen:

- Wie verhält sich die Qualität von R-EED-Resultaten im Vergleich zu C-EED-Resultaten bei fester Kompressionsrate?
- Wie groß ist der Qualitätsgewinn, der nur durch Ausnutzung des Speichergewinns durch Elimination redundanter Dateikopfdaten und ganzheitlicher Entropie-Codierung gegenüber R-EED erzielt wird? (Vergleich zwischen C-EED2D und R-EED)
- Welcher Qualitätsgewinn entsteht nur durch die Ausnutzung dreidimensionalen Inpaintings im Vergleich zu 2D-Schnittbild-Inpainting? (Vergleich zwischen C-EED2D und C-EED)
- Wie verhalten sich die EED-basierten Methoden im Vergleich zu JPEG2000?

Da diese Arbeit als Machbarkeitsstudie aufzufassen ist, die das Augenmerk auf die erreichbare Qualität für mittlere bis hohe Kompressionsstufen legt, werden keine Laufzeitanalysen durchgeführt. Lediglich Vergleichswerte für C-EED2D und R-EED werden genannt, um die Signifikanz des Laufzeitgewinns durch globale Parameteroptimierung in C-EED2D und Schnittbild-Parameteroptimierung in R-EED zu illustrieren.

Aufgrund ebendieser Laufzeitunterschiede zwischen R-EED und C-EED2D wurden mehr vergleichende Experimente mit C-EED2D und C-EED durchgeführt als mit R-EED, nachdem nachgewiesen wurde, dass C-EED2D auf allen untersuchten Testbildern qualitativ bessere Ergebnisse liefert als R-EED.

Neben den bereits genannten Kernexperimenten wurden weitere Versuche durchgeführt, welche anhand der folgenden Fragen Rückschlüsse auf die Auswirkungen einzelner Kompressionsschritte zulassen sollen und damit einer genaueren Analyse der Qualitätsvergleiche dienlich sind:

- Wie stark unterscheiden sich C-EED2D und C-EED bezüglich der Verteilung von Fehlern im komprimierten Bild?
- Welche dreidimensionalen Punktmuster des C-EED-Verfahrens eignen sich am Besten für verschiedene Testbilder?
- Wie ist der Qualitätsgewinn durch Grauwertoptimierung und Maskeninversion einzuschätzen?
- Wie groß ist der mögliche Qualitätsgewinn durch ROI-Codierung auf klassischen Testbildern?

Die konkrete Ausgestaltung der Experimente, die mit dem Ziel der Beantwortung der genannten Fragen durchgeführt wurden, ist im folgenden Abschnitt 5.2 zu finden.

5.2 Testumgebung

5.2.1 Hard- und Software

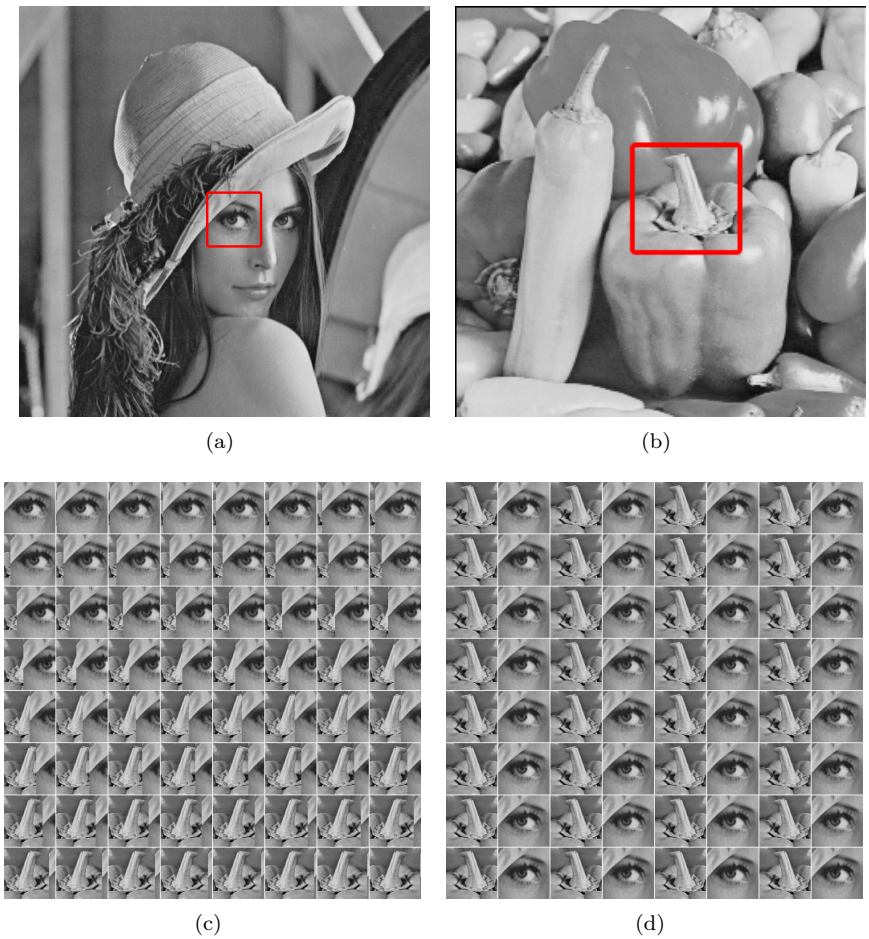
Alle vorliegenden Tests wurden auf Linux-basierten Systemen mit der Referenzimplementierung von C-EED und C-EED2D durchgeführt. Die Tests mit R-EED basieren auf einer modifizierten Referenzimplementierung, welche die gleiche EED-Diskretisierung wie C-EED2D einsetzt. Als Compiler wurde ausschließlich *gcc Version 4.1.2* mit den Compileroptionen *-O 3* und *-std=C99* eingesetzt. Da keine Laufzeittests durchgeführt wurden, wurden Tests auf mehreren getrennten Systemen mit unterschiedlicher Hardwarekonfiguration durchgeführt. Dabei wurde verifiziert, dass die Ergebnisse nicht durch unterschiedliche Rechengenauigkeit verfälscht wurden.

Die Testergebnisse von JPEG2000 wurden mit dem Programm convert[1] ermittelt. Da JPEG2000 bei Schnittbildern mit niedriger Auflösung (64×64) aufgrund der Größe des Dateikopfes so stark zurückfällt, dass ein Vergleich nicht mehr möglich ist, werden modifizierte Testbilder verwendet. Je acht der 64 Schnittbilder werden hierzu nebeneinander gesetzt und ergeben in acht Zeilen ein einzelnes quadratisches Bild der Dimension 512×512 (siehe z.B. Abb. 5.1(c)). Diese Form der Darstellung des 3D-Bildes behebt nicht nur die Probleme von JPEG und JPEG2000, sondern erlaubt 2D-Kompressionsmethoden auch die Codierungs-Redundanz auszunutzen, sowie möglicherweise auch Redundanzen zwischen benachbarten Schnittbildern zu beachten.

5.2.2 Testbilder

In diesem Kapitel werden die Ergebnisse von Kompressionsexperimenten mit Testbildern zweier Grundtypen verwendet. Zum einen wurden synthetische Testbilder konstruiert, welche die gleiche Gitterweite in allen drei Dimensionen aufweisen und speziell konstruiert wurden, um Erwartungen zu überprüfen. Zum anderen wurden Tests auf realen medizinischen 3D-Daten durchgeführt.

[1] Verfügbar unter `http://www.imagemagick.org`



(a) (b)

(c) (d)

Abbildung 5.1: **Synthetische Testbilder.** Die Abbildung (a) und (b) zeigen die Originalbilder *Lenna* und *Peppers*, in denen die beiden für die Erstellung synthetischer Bilder verwendeten Bildausschnitte der Dimension 64×64 durch einem roten Rahmen markiert sind. Die daraus entstehenden Bildsequenzen *transition.pdm* (Abb. (c)) und *interleaved.pdm* (Abb. (d)) sind unter den Originalen abgebildet. Dabei wurden die Bilder in der durch die Sequenz vorgegebenen Reihenfolge zeilenweise von links nach rechts hintereinander gesetzt, um die Sequenz durch ein zweidimensionales Bild der Dimension 512×512 (exklusive Rahmen) darzustellen. Dieses Format wird auch für die Tests mit JPEG2000 eingesetzt, allerdings ohne die weißen Rahmen.

Synthetische Testbilder

Die drei hier behandelten synthetischen Testbilder haben alle die gleiche Dimension $64 \times 64 \times 64$ und sind aus je einem Ausschnitt der klassischen 2D-Testbilder *Lenna*[2] (Abb. 5.1(a)) und *Peppers*[3] (Abb. 5.1(b)) zusammengesetzt.

[2] 8-bit pgm-Version, verfügbar unter http://www.ece.rice.edu/~wakin/images/
[3] Verfügbar unter http://decsai.ugr.es/~javier/denoise/

extended.pdm : Der Ausschnitt aus dem Lenna-Testbild wird 64-Mal unverändert als x-y-Schnittbild hintereinander gesetzt. In der dritten Dimension ist das Bild daher vollständig redundant, weshalb ein signifikanter Qualitätsgewinn sowohl durch ganzheitliche Entropie-Codierung in C-EED2D, als auch durch dreidimensionales Inpainting in C-EED erwartet wird.

transition.pdm : In diesem Testbild wird ein einfacher Videoschnitt-Effekt simuliert. Der Ausschnitt aus dem Lenna-Testbild definiert das erste Schnittbild bzw. das erste Bild der Bildsequenz. In den darauffolgenden Schnittbildern wird dieser Ausschnitt jeweils durch eine Translation um einen Bildpunkt pro Bild in positiver x-Richtung verschoben. Beginnend mit Bild 2 wird in gleicher Weise der Peppers-Bildausschnitt vom linken Bildrand ($y = 1$) aus "ins Bild geschoben" (siehe Abbildung 5.1(c)). Für diese Bildsequenz werden aufgrund der vorhandenen Redundanzen in der dritten Dimension gute Ergebnisse für C-EED und C-EED2D erwartet, die allerdings nicht dem Idealfall aus dem *extended*-Experiment entsprechen.

interleaved.pdm Das dritte synthetische Testbild besteht aus einer Bildsequenz, die aus den sich abwechselnden Ausschnitten der Lenna- und Peppers-Testbilder besteht (siehe Abbildung 5.1(d)). Diese Sequenz kann als periodische Erweiterung des *transition*-Testbildes mit sehr niedriger Abtastrate in der Zeitdimension aufgefasst werden. Dieses Testbild ist so konstruiert, dass zwei aufeinanderfolgende Bilder keinen Zusammenhang in der dritten Dimension aufweisen. Aufgrund dieses Umstandes werden schlechte Ergebnisse mit 3D-Inpainting in C-EED erwartet. *interleaved* bildet damit das Gegenstück zu dem Idealfall *extended*.

Reale Datensätze

Neben den synthetischen Testbildern werden reale medizinische Datensätze aus bildgebenden Verfahren eingesetzt. Aufgrund der hohen Laufzeit der Parameteroptimierung wurden hauptsächlich würfelförmige Ausschnitte der Dimension $64 \times 64 \times 64$ aus den Bildern *trab.pdm*[4] und *bone.pdm*[5] in Abbildung 5.2 bzw. 5.3 anstatt der Originale verwendet. Die Bildausschnitte werden mit *bone64A*, *bone64B* und *brain64* bezeichnet.

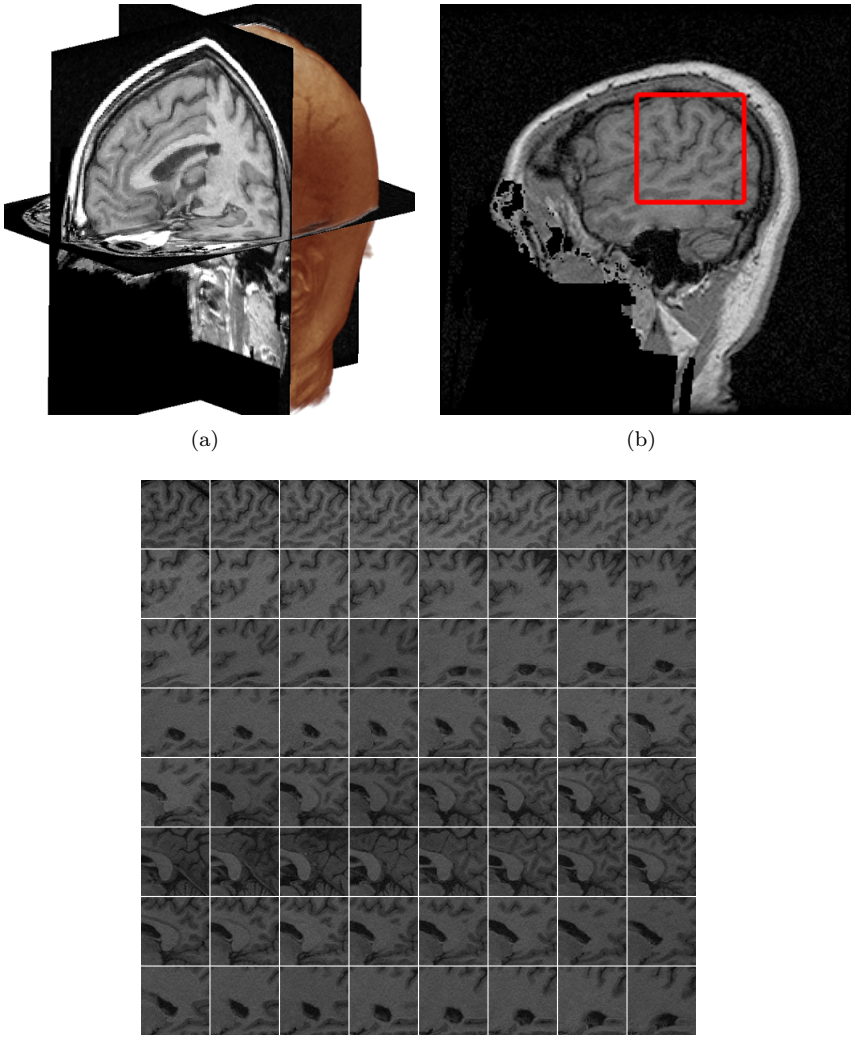
Dabei wurden die drei hier vorgestellten Bildausschnitte so gewählt, dass sie über unterschiedliche Eigenschaften verfügen: *bone64A* enthält signifikante Anteile eines bis auf Bildstörungen einförmigen Hintergrundes, während *bone64B* sowohl über einförmige, durch scharfe Kanten abgegrenzte Bildregionen, als auch über unregelmäßige Bildteile verfügt. Beide Ausschnitte enthalten streifenförmige Artefakte. *brain64* hingegen enthält zwar ebenfalls mit bloßem Auge klar identifizierbare, teils filigrane Teilbereiche ähnlichen Grauwerts, diese sind jedoch durch Bildrauschen beeinträchtigt.

Weitere Tests wurden auf einem 3D-Bild mit voller x-y-Auflösung (256×256) durchgeführt, welches aus den ersten vierundsechzig Bildern des Knochen-Scans *trab.pdm* besteht. Dieses Testbild erhält den Namen *bone*.

Bei den medizinischen Testbildern ist zu beachten, dass die Gitterweite in z-Dimension doppelt so groß ist wie in x- bzw. y-Dimension. Weiterhin wurden

[4] Intern verwendetes Testbild der Mathematical Image Analysis Group

[5] DICOM-Beispieldatensatz des Visualisierungsprogrammes Slicer, verfügbar unter http://www.slicer.org/slicerWiki/images/5/51/Slicer3MinuteDataset.zip

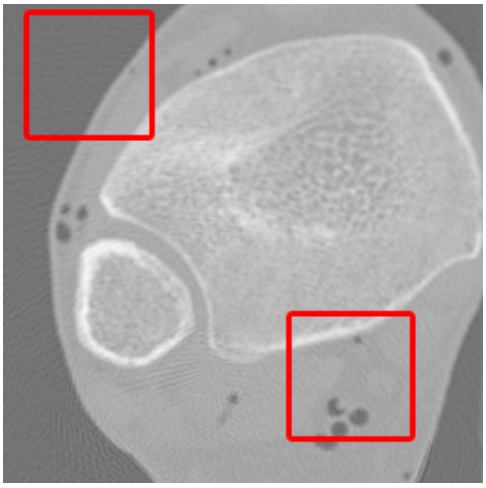


(a) (b)

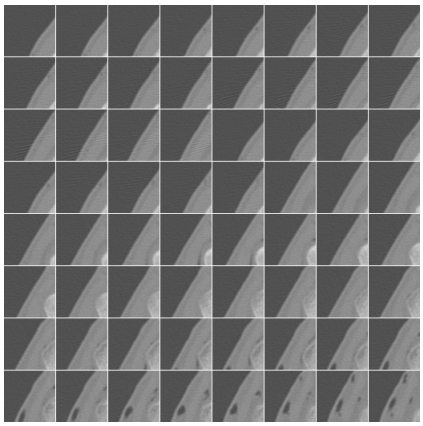
(c)

Abbildung 5.2: **Medizinische Testbilder: Gehirn-Scan.** Da dreidimensionale Darstellungen wie in Abb. (a) durch Verzerrung Bildvergleiche erschweren, werden die Datensätze *trab.pdm* (Abb. (a)) und *brain.pdm* (Abb. (b)) durch zweidimensionale Schnittbilder repräsentiert. Der rote Rahmen deutet an, wie der würfelförmigen Bildausschnitt *brain64* (Abb. (c)) in x- und y-Richtung eingeschränkt wird.

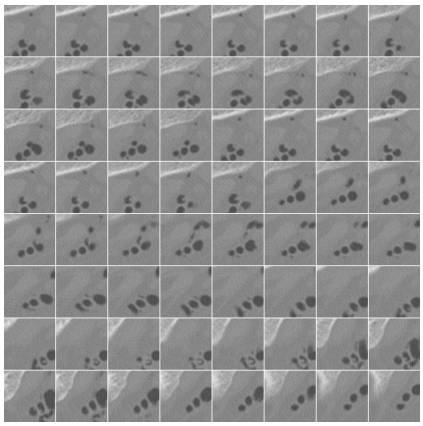
neben den hier beschriebenen Testbildern noch auf anderen Bildern Experimente durchgeführt, deren Ergebnisse jedoch zu keinen neuen Erkenntnissen führten. Aus Gründen der Übersichtlichkeit wird lediglich eine repräsentative Auswahl an Testbildern vorgestellt.



(a)



(b)



(c)

Abbildung 5.3: **Medizinische Testbilder: Knochen-Scan.** Abbildung (a) zeigt das zwanzigste x-y-Schnittbild der Testdatei *trab.pdm*. Die roten Rahmen deuten an, wie die würfelförmigen Bildausschnitte *bone64A* (Abb. (b)) und *bone64B* (Abb. (c)) in x- und y-Richtung eingeschränkt werden.

5.2.3 Testtypen

Um die in Abschnitt 5.1 aufgeworfenen Fragen zu beantworten, werden verschiedene Arten von Kompressionsexperimenten mit den Bildern aus Abschnitt 5.2.2 durchgeführt. Grauwertoptimierung und Umkehrung der Inpaintingmaske werden von C-EED, C-EED2D und R-EED, sofern nicht ausdrücklich erwähnt, laufzeitbedingt nicht eingesetzt.

Als Vergleichskriterium wird jeweils der mittlere quadratische Fehler (MSE, engl. *mean square error*) zwischen dem komprimiertem Bild und dem zugehörigen Original verwendet. Der MSE zweier Bilder u und f der Dimension

$n_x \times n_y \times n_z$ ist definiert durch:

$$\mathrm{MSE}(u, f) := \sum_{(i,j,k)\in\Omega} \frac{(u_{i,j,k} - f_{i,j,k})^2}{n_x n_y n_z} \tag{5.1}$$

Punktmustervergleich: Um die Vermutungen über den Einfluss der für die adaptive Quaderunterteilungen in C-EED eingesetzten Punktmuster zu verifizieren, werden auf synthetischen und realen 3D-Testdaten Kompressionen mit 14 verschiedenen Punktmustern und festem Kompressionsverhältnis 20:1 durchgeführt. Dabei werden jeweils die Parameter λ, q, a und l für jedes einzelne Punktmuster optimiert. Zur Entropie-Codierung wird einheitlich statische Bereichscodierung verwendet.

Codierungs-Redundanztests Für einige Testbilder mit festem Kompressionsverhältnis 20:1 werden Vergleiche zwischen R-EED und C-EED2D mit Optimierung der Parameter λ, q, a und l, sowie statischer Bereichscodierung angestellt.

Qualitätsvergleich: Für Kompressionsraten von 10:1 bis 100:1 werden komprimierte Bilder mit Kompressionsraten in der Nähe der Zehnerschritte (10:1, 20:1 usw.) unter Nutzung der Algorithmen C-EED2D, C-EED und JPEG2000 angefertigt. C-EED2D und C-EED nutzen dabei eine volle Parameteroptimierung für die Parameter λ, q, a und l. Die Punktmuster werden gemäß der besten Ergebnisse aus den Punktmustervergleichen ausgewählt (Muster AB oder AD). Die Ergebnisse werden anhand des MSE verglichen.

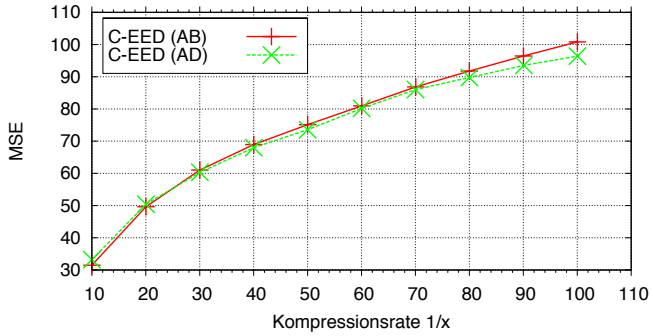
Fehlerverteilung: Bei festem Kompressionsverhältnis 20:1 und Parameteroptimierung von λ, q, a und l wird für C-EED2D, C-EED, R-EED und JPEG2000 die Fehlerverteilung in den einzelnen Schnittbildern untersucht, um eine eingehendere Analyse der Ergebnisse aus den Qualitätsvergleichen zu ermöglichen.

Grauwertoptimierung: Aufgrund der langen Laufzeit der Grauwertoptimierung wird an lediglich zwei Beispielen exemplarisch demonstriert, wie der MSE bei konstanter Kompressionsrate durch Grauwertoptimierung verbessert werden kann. Dabei werden ein Grauwertoptimierungsschritt, eine nachfolgende EED-Parameteroptimierung sowie eine Maskeninversion durchgeführt. Die restlichen Parameter werden anhand der Optimierungsergebnisse aus den Qualitätsvergleichen fest gewählt.

ROI-Codierung: Als Machbarkeitsstudie wird an dem 2D-Testbild Lenna demonstriert, welche Auswirkungen ROI-Codierung mit R-EED auf den Fehler innerhalb der ROI hat.

5.3 Ergebnisse und Analyse

In den folgenden Abschnitten werden die Ergebnisse der durchgeführten Experimente beschrieben und analysiert. Eine zusammenfassende Diskussion der gewonnenen Erkenntnisse erfolgt in Abschnitt 5.4.



(a)

Abbildung 5.4: **Punktmuster bei veränderlicher Kompressionsrate.** Für niedrige Kompressionsraten liefert das Punktmuster AB qualitativ leicht hochwertigere Rekonstruktionen des Bildes *brain64* als das Punktmuster AD. Diese Rangfolge ändert sich bei wachsender Kompressionsrate. Insbesondere für hohe Kompressionsraten ab 70:1 steigt der MSE des Musters AB schneller an als der von AD.

5.3.1 Punktmustervergleich

In Tabelle 5.1 werden Ergebnisse für Punktmusterexperimente mit den Bildern *transition*, *interleaved*, *extended* und *bone64A* dargestellt. Das Kompressionsverhältnis ist dabei auf 20:1 (mit Abweichungen kleiner eins) fixiert. Betrachtet man die Ergebnisse für die verschiedenen Bilder getrennt, so ist festzustellen, dass der MSE für verschiedene Punktmuster stark variiert. Die Ausprägung dieser Variationen sind von Bild zu Bild verschieden, insbesonders ändert sich auch die MSE-Rangfolge in Abhängigkeit vom Inhalt des Bildes. Die Punktmuster AB und AD stellen sich dabei für die meisten Experimente in Tabelle 5.1 und weiteren durchgeführten Tests als jeweils beste oder zweitbeste Wahl heraus.

Wie erwartet erzielen Punktmuster, die den Mittelpunkt enthalten, in der Regel die besten Ergebnisse. Allerdings zeigen die Ergebnisse auch, dass die Inklusion des Mittelpunktes nicht zwangsläufig für alle Kombinationen von Muster und Bildinhalten Vorteile bringt. Wie erwartet bietet die Verwendung der Quadereckpunkte B eine gute Verteilung der Maskenpunkte im Bild, vor allem in Kombination mit dem Mittelpunkt A. Die Kantenmittelpunkte D liefern konsistent bessere Ergebnisse als die Mittelpunkte der Seitenflächen C. Bemerkenswert ist hierbei insbesondere auch, dass sich das Muster AD in fast allen Tests mit der Kompressionsrate 20:1 als beste Wahl erweist. Beide Muster setzen sich auch gegen die adaptive Wahl E und F durch.

Weitere Tests zeigen, dass die qualitative Rangfolge der Punktmuster nicht nur vom Bildinhalt abhängt, sondern auch, wenngleich in geringerem Maße, mit Änderungen im Kompressionsverhältnis variiert. Abbildung 5.4 demonstriert, wie sich die Rangfolge der favorisierten Muster AD und AB bei steigendem Kompressionsverhältnis verändert.

Zusammenfassend lässt sich aus den Punktmusterexperimenten schließen, dass für eine vollständige Optimierung des MSEs auch die Wahl des Punktmusters an Bildinhalt und Kompressionsverhältnis angepasst werden muss. Aufgrund der teilweise stark variierenden Rangfolge sind exakte Vorhersagen des

	transition	interleaved	extended	bone64A
A	147.6840	982.7067	47.2649	7.602993
C	189.8983	852.9241	129.4922	18.386349
D	121.2733	874.7211	36.3716	6.597172
E	133.1494	1046.4601	42.1172	6.832485
F	125.3971	1542.5681	40.0984	9.868942
AB	112.1352	926.6998	**30.9097**	**6.329212**
AC	198.1832	865.3415	121.282154	17.244217
AD	**111.0942**	**801.2542**	32.431503	6.432735
AE	137.9564	877.855	50.341698	7.570663
AF	118.2150	974.4872	35.514118	6.627331
BC	116.4277	876.1126	34.214340	6.531384
BD	159.7889	853.6154	90.614803	17.129925
BE	188.7748	1536.2098	32.637096	9.648579
BF	114.5914	981.6726	40.128819	6.592529

Tabelle 5.1: **MSE: Punktmustervergleich.** Einer Auswahl verschiedener Punktmuster wird der MSE der Bilder *transition*, *interleaved*, *extended* und *bone64A* mit einer Kompressionsrate von 20:1 zugeordnet. Das beste Ergebnis pro Bild ist fett hervorgehoben, das zweitbeste unterstrichen.

bestgeeigneten Punktmusters schwierig, allerdings bieten die Punktmuster AB und AD konsistent sehr gute Werte. Um einen Laufzeitgewinn zu erzielen, kann sich der Algorithmus in der Regel ohne signifikante MSE-Verluste auf diese beiden Muster beschränken.

5.3.2 Codierungs-Redundanztests

Auf den Bildern *extended*, *interleaved* und *bone64A* mit Kompressionsverhältnis 20:1 erreicht C-EED2D auf allen Bildern einen um ca. acht bis zwanzig Prozent niedrigeren MSE als R-EED (siehe Tabelle 5.2). Dabei ist die Kompressionsrate ohne Entropie-Codierung bei C-EED2D um sechzehn bis fünfundzwanzig Prozent geringer als bei R-EED (siehe Tabelle 5.3).

Auf den untersuchten Testbildern reicht der Speicherplatzgewinn bei C-EED2D durch ganzheitliche Entropie-Codierung und Elimination von Dateikopfinformationen aus, um gegenüber R-EED auf allen Testbildern einen Qualitätsgewinn zu erreichen.

	bone64A	extended	interleaved
R-EED	8,1308	104,4938	222,7966
C-EED2D	7,4466	82,8275	185,9578

Tabelle 5.2: **MSE: Codierungs-Redundanztests.** Für die Bilder *interleaved*, *extended* und *bone64A* wird der MSE bei fester Kompressionsrate 20:1 betrachtet.

	bone64A	extended	interleaved
R-EED	13,7982:1	13,7233:1	13,0995:1
C-EED2D	11,5207:1	10,3304:1	9,8178:1

Tabelle 5.3: **Kompressionsverhältnis ohne Entropie-Codierung.** Für die Bilder *interleaved*, *extended* und *bone64A* wird die Kompressionsrate ohne Entropiecodierung betrachtet, die vom jeweiligen Algorithmus benötigt wird, um mit Entropiecodierung eine Kompressionsrate von 20:1 zu erreichen.

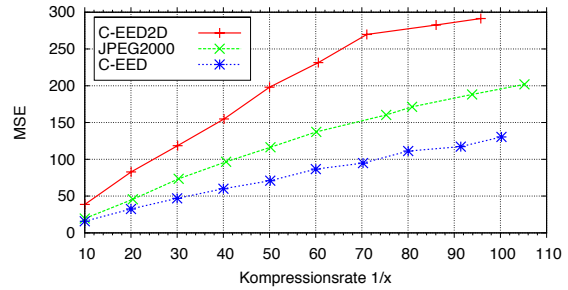
5.3.3 Qualitätsvergleiche

Bei den Qualitätstests auf den synthetischen Bildern *extended* und *transition* erzielt C-EED für nahezu alle untersuchten Kompressionsraten bessere Ergebnisse als C-EED2D und JPEG2000 (siehe Abb. 5.5(a) und 5.5(b)). Der Qualitätsgewinn durch dreidimensionales Inpainting beträgt gegenüber den C-EED2D-Ergebnissen zwischen vierzig und sechzig Prozent des C-EED2D-MSE. Bei JPEG2000 betragen diese Gewinne bis zu ca. dreißig Prozent, allerdings ist JPEG2000 für niedrige Kompressionsraten im Testbild *transition* dem C-EED-Algorithmus überlegen. C-EED2D liegt auf diesen Bildern ohne Grauwertoptimierung für alle Kompressionsraten qualitativ deutlich hinter JPEG2000.

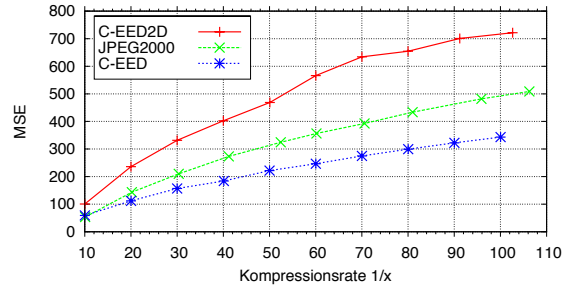
Im Experiment *interleaved* übersteigen die MSE-Werte von C-EED die Ergebnisse der C-EED2D- und JPEG2000-Tests um mehr als das Vierfache und C-EED liegt qualitativ konsistent hinter JPEG2000 (siehe Abb. 5.5(c)).

Die ausgewählten praktischen Experimente decken ein breites Spektrum an beobachteten MSE-Rangfolgen ab (Abb. 5.6(a) bis 5.6(c)). Auf dem Testbild *bone64A* sind sowohl C-EED2D als auch C-EED JPEG2000 für nahezu alle Kompressionsraten deutlich überlegen, in *bone64B* zeigt sich dieses Muster nur für hohe Kompressionsraten, während bei niedriger Rate JPEG2000 qualitativ hochwertigere Ergebnisse erzielt. Im Experiment mit dem Testbild *brain64* erreicht C-EED nur für sehr hohe Kompressionsverhältnisse Ergebnisse, die mit JPEG2000 konkurrieren können. Das Verhältnis von C-EED zu C-EED2D hingegen bleibt jedoch weitgehend konsistent. Für niedrige Kompressionsraten liegen die Fehlerwerte nah beieinander, mit steigender Kompressionsrate erzielt C-EED jedoch deutlich bessere Ergebnisse. Ausgewählte Vergleichsbilder zu den durchgeführten Tests sind in Abbildung 5.7 und Anhang A zu finden.

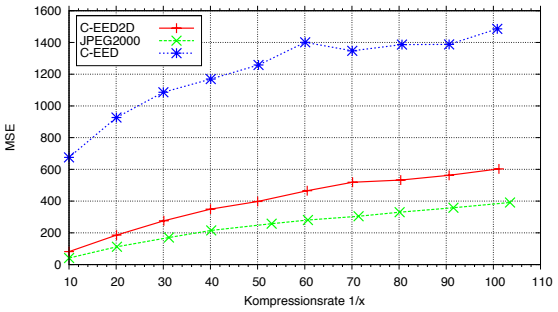
Für das Teilbild *bone* mit Dimension $256 \times 256 \times 64$ wurden lediglich Tests mit JPEG2000 und C-EED für eine feste Kompressionsrate von ca. 20:1 durchgeführt. Dabei erreicht C-EED ohne Grauwertoptimierung einen Fehler von ca.



(a) extended



(b) transition

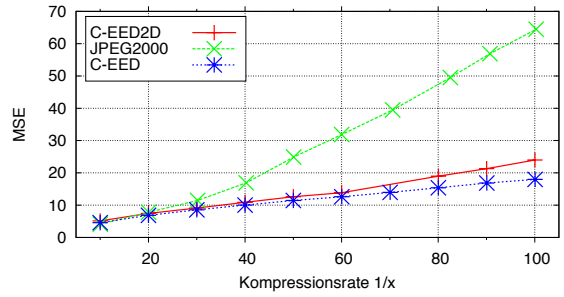


(c) interleaved

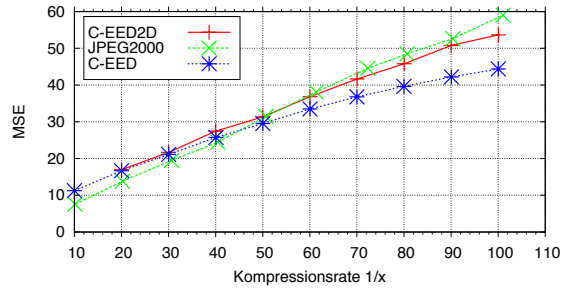
Abbildung 5.5: **Qualitätsvergleich: synthetische Bilder.** Die obenstehenden Diagramme zeigen den Zusammenhang zwischen Kompressionsrate (x-Achse) und MSE (y-Achse) für die Algorithmen C-EED2D, C-EED und JPEG in Bezug auf das jeweilige synthetische Testbild. C-EED und C-EED2D nutzen hierbei keine Grauwertoptimierung und Maskeninversion.

12, 83. Der MSE von JPEG2000 liegt bei 14, 10 (siehe auch Abbildung 5.8).

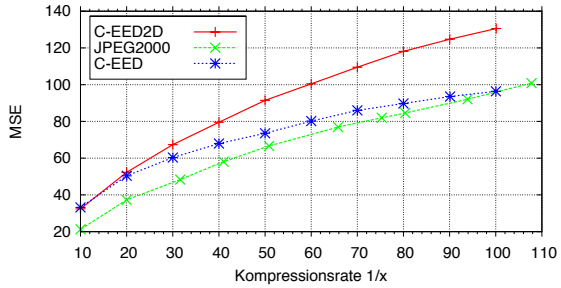
Die Ergebnisse des Qualitätsvergleichs für synthetische Bilder entsprechen im Wesentlichen den bei der Konstruktion der Bilder gestellten Erwartungen. C-EED kann die bewusst erzeugten Redundanzen *transition* und *extended* nutzen, um deutliche Qualitätsverbesserungen gegenüber den zweidimensionalen Fällen zu erreichen. Die Experimente mit *interleaved* hingegen zeigen wie erwartet Schwachstellen im C-EED-Verfahren auf. Derartige Extremfälle treten zwar in Realdaten nicht auf, in Abschnitt 5.3.4 werden aber dennoch die Gründe dieses Problems analysiert und mögliche Auswirkungen auf praktische Anwendungen



(a) bone64A



(b) bone64B

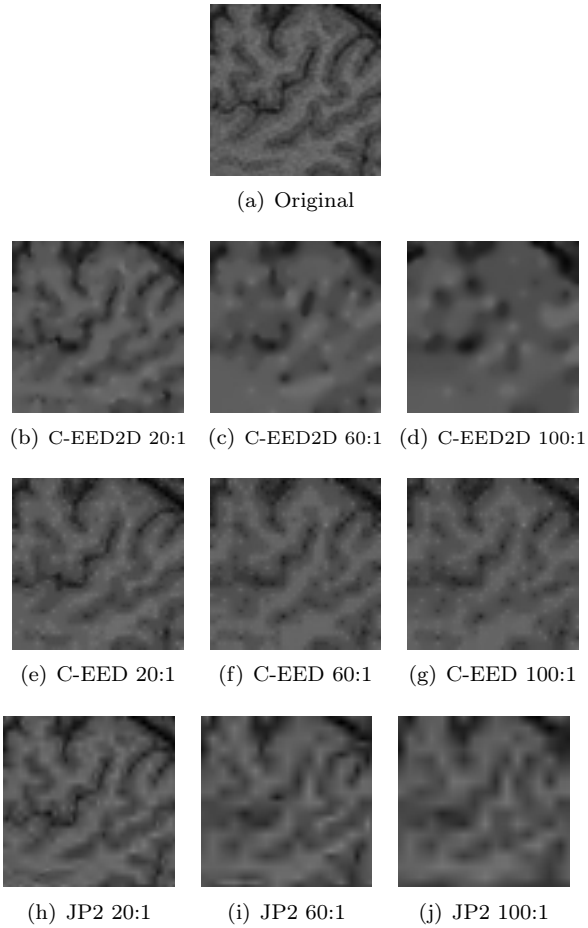


(c) brain64

Abbildung 5.6: **Qualitätsvergleich: Realdaten.** Die obenstehenden Diagramme zeigen den Zusammenhang zwischen Kompressionsrate (x-Achse) und MSE (y-Achse) für die Algorithmen C-EED2D, C-EED und JPEG in Bezug auf das jeweilige medizinische Testbild. C-EED und C-EED2D nutzen hierbei keine Grauwertoptimierung und Maskeninversion.

untersucht.

Auf Realdaten kann C-EED ohne Grauwertoptimierung und Maskeninversion sowohl in dem Test mit voller x-y-Auflösung, als auch in einigen der Teilbildtests niedrigere Fehlerwerte vorweisen als JPEG2000. Im visuellen Direktvergleich macht sich der niedrigere MSE vor allem bei hohen Kompressionsraten deutlich bemerkbar, bei moderaten Kompressionsraten werden im Vergleich zu JPEG2000 in erster Linie mehr Details und schärfere Kanten von C-EED erhalten. In Abschnitt 5.3.5 wird untersucht, inwiefern die Ergebnisse von C-EED2D durch Grauwertoptimierung weiter verbessert werden können.



(a) Original

(b) C-EED2D 20:1 (c) C-EED2D 60:1 (d) C-EED2D 100:1

(e) C-EED 20:1 (f) C-EED 60:1 (g) C-EED 100:1

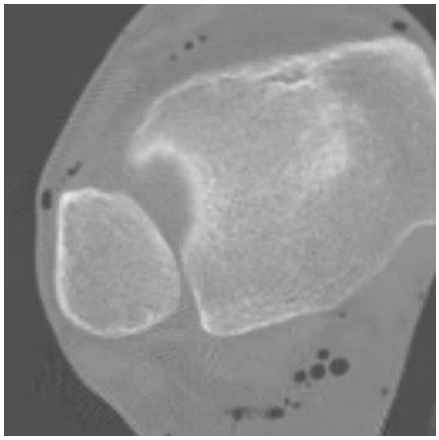
(h) JP2 20:1 (i) JP2 60:1 (j) JP2 100:1

Abbildung 5.7: **Visueller Vergleich: brain64.** Die obigen Abbildungen zeigen Kompressionsergebnisse der Algorithmen C-EED, C-EED2D und JPEG2000 (Abgekürzt zu JP2) angewendet auf das vierte Schnittbild von *brain64*. C-EED und C-EED2D nutzen jeweils das Muster AB, Parameteroptimierung von λ, l, a und q. Grauwertoptimierung und Maskeninversion wurden nicht durchgeführt.

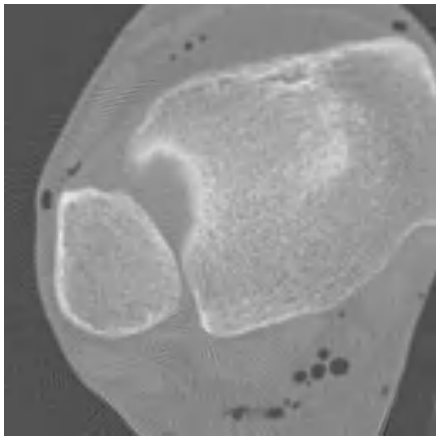
5.3.4 Fehlerverteilung

In Abbildung 5.9(a) ist die Fehlerverteilung für C-EED2D und C-EED pro x-y-Schnittbild des Testbildes *transition* bei einem Kompressionsverhältnis von 20:1 dargestellt. Bei C-EED2D nimmt der MSE mit wachsender Schnittbildrate erkennbar zu, auch wenn keine monotone Erhöhung des Fehlers stattfindet.

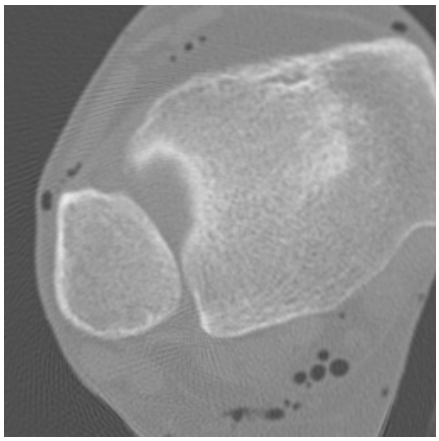
Für C-EED ist ebenfalls eine leichte Tendenz zur Fehlerzunahme zu erkennen, allerdings ist diese weniger ausgeprägt als bei C-EED2D. Zudem dominiert ein sich wiederholendes Muster den MSE aufeinanderfolgender Schnittbilder. Die Schnittbilder mit ungerader Schnittbildnummer weisen einen größeren Fehler auf als die mit gerader Nummer. Unter den geraden Schnittbildern lässt sich überdies ebenfalls ein festes Muster erkennen: beginnend mit dem ersten Bild wechselt sich jeweils ein gerades Schnittbild mit niedrigem und ein Schnitt-



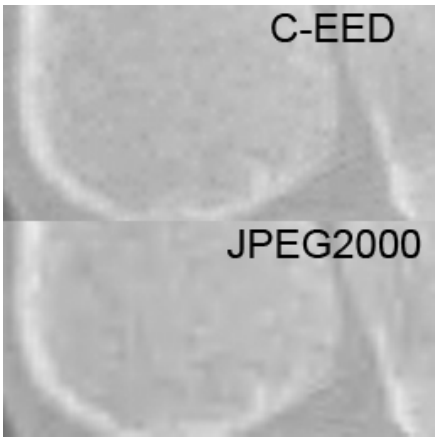
(a) C-EED



(b) JPEG2000



(c) Original

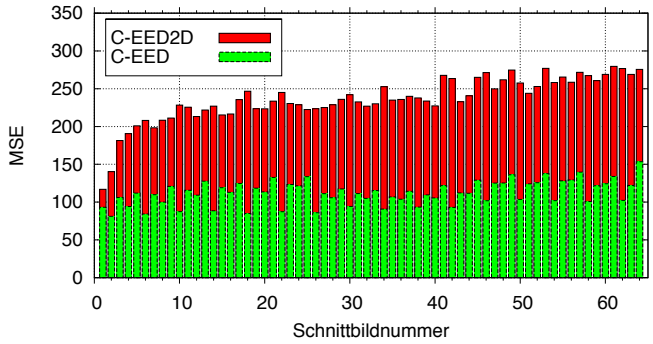


(d) Vergrößerung

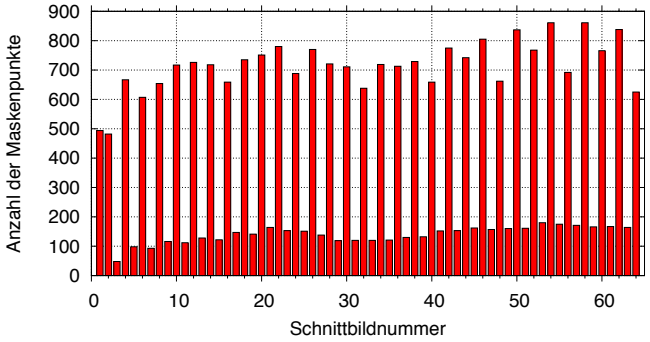
Abbildung 5.8: **Qualitätsvergleich: Testbild bone.** Das Kompressionsergebnis von C-EED (Abb. (a)) weist einen MSE von 12,831064 auf, während die JPEG2000-Version des Bildes (Abb. (b)) bei gleicher Kompressionsrate von 20:1 einen Fehlerwert von 14,098142 erreicht. Abgebildet ist jeweils das zwanzigste x-y-Schnittbild beider Kompressionsergebnisse. Die Parameter für die C-EED Kompression sind hierbei $\lambda = \frac{1}{59}$, $q = 36$, $a = 1,167278$, $l = 1,595215$ mit Punktmuster AB und ohne Grauwertoptimierung oder Maskeninversion. Abb. (d) dient der Verdeutlichung der Qualitätsunterschiede: C-EED erhält mehr Details und schärfere Kanten als JPEG2000.

bild mit höherem MSE ab. Dabei bleibt der höhere MSE jeweils unter dem der ungeraden Schnittbilder.

Dieses Verhalten von C-EED2D ergibt sich aus der Maskenerstellung durch adaptive Quaderunterteilung. Betrachtet man die Verteilung von Maskenpunkten in den einzelnen Schnittbildern (vgl. Abb. 5.9(b)), so stellt sich heraus, dass die geraden Bilder jeweils wesentlich mehr Maskenpunkte aufweisen als die ungeraden. Die Anzahl der Maskenpunkte beeinflusst auch bei 3D-Inpainting



(a) Fehlerverteilung



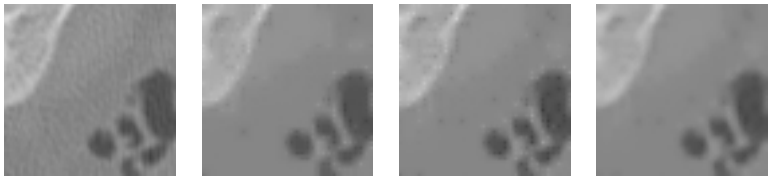
(b) Verteilung der Maskenpunkte

Abbildung 5.9: **Fehlerverteilung in transition.pdm.** Diagramm (a) zeigt einen Vergleich der Fehlerverteilung für die einzelnen Schnittbilder des mit C-EED bzw. C-EED2D komprimierten Testbildes *transition.pdm*. Dabei bezeichnet jeder Balken die Ergebnisse für ein Schnittbild, die Höhe repräsentiert den MSE. In Abbildung 5.9(b) wird auf gleiche Weise die Anzahl der Maskenpunkte pro Schnittbild für C-EED im gleichen Bild dargestellt.

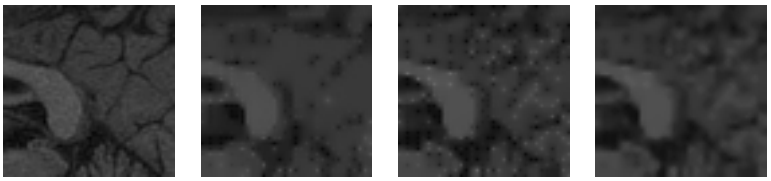
den Rekonstruktionsfehler im jeweiligen Schnittbild und führt zu den bereits beobachteten Mustern in der Fehlerverteilung. Für höhere Kompressionsraten und unregelmäßigere Bildinhalte ergeben sich andere und weniger regelmäßige Muster in der Fehlerverteilung, allerdings bleiben die Maskenpunkte aufgrund der verwendeten Punktmuster weiterhin ungleichmäßig über die Schnittbilder verteilt.

Bei C-EED2D sind derartige Muster nicht zu beobachten, da für jedes Schnittbild implizit eine Quaderunterteilung vorgenommen wird. Keines der Schnittbilder wird dabei durch die Punktmuster bevorzugt, lediglich in der x- und y-Dimension lassen sich derartige Regelmäßigkeiten in der Fehlerverteilung beobachten. Unterschiede im MSE der Schnittbilder ergeben sich allein anhand des Inhaltes der Bilder. Im Beispiel *transition* rührt der Anstieg des Fehlers daher, dass mit wachsender Schnittbildnummer der *Peppers*-Ausschnitt größeren Anteil am Gesamtbild hat, welcher sich mit EED-Inpainting nur mit niedrigerer Qualität als der Bildausschnitt *Lenna* komprimieren lässt.

Auch wenn die unregelmäßige Verteilung des MSE als Nachteil von C-EED erscheint, ist zu beachten, dass durch die 3D-Bildunterteilung der Einfluss der



(a) Original (b) MSE: 21,1099 (c) MSE: 18,1381 (d) MSE: 17,7758



(e) Original (f) MSE: 96.4661 (g) MSE: 72.6227 (h) MSE: 68,4205

Abbildung 5.10: **Visueller Vergleich: Grauwertoptimierung.** Die obigen Abbildungen zeigen Kompressionsergebnisse mit C-EED unter Anwendung auf die Bilder *bone64B* (Kompressionsverhältnis 30:1) und *brain64* (Kompressionsverhältnis 90:1). In den Abbildungen (a) und (e) ist jeweils das Schnittbild Nr. 56 des Originals zu sehen. Abb. (b) und Abb. (f) wurden jeweils ohne Grauwertoptimierung und Maskeninversion erstellt. Die Abbildungen (c) und (g) zeigen das Ergebnis nach zusätzlicher Grauwertoptimierung, in Abb. (d) und (h) wurde ein zusätzlicher Maskeninversionsschritt durchgeführt. Bei *brain64* ist der Qualitätsgewinn durch Grauwertoptimierung in den feinen Strukturen der Gehirnwindungen deutlicher nachzuvollziehen als die Änderungen in *bone64B*. Die Maskeninversion verbessert jedoch die wahrgenommene Bildqualität in beiden Testfällen deutlich.

Schnittbildinhalte auf den Fehler abgemildert wird und die Abweichung des MSEs zwischen verschiedenen Schnittbildern insgesamt im Vergleich zu C-EED-2D sinkt. Lediglich auf Bildern, bei denen kein Zusammenhang zwischen aufeinanderfolgenden Schnittbildern in der z-Dimension besteht oder die Abtastrate in z-Richtung gering ist, wird die Verteilung der Maskenpunkte zu einem signifikanten Problem für C-EED, da eine Rekonstruktion der Schnittbilder mit wenigen Maskenpunkten aus den Informationen benachbarter Schnittbilder nicht oder nur ungenau erfolgen kann. Dieser Effekt wird in sehr starker Ausprägung durch das Experiment *interleaved* belegt.

5.3.5 Grauwertoptimierung

Aufgrund der hohen Laufzeit der Grauwertoptimierung wurden lediglich Stichproben für Testbilder angefertigt, bei denen JPEG2000 bei gleicher Kompressionsrate eine höhere Qualität als C-EED ohne Grauwertoptimierung aufweist. Als Repräsentanten für moderate bzw. hohe Kompressionsverhältnisse werden im Folgenden die Ergebnisse für die Testbilder mit *bone64B* mit dem Verhältnis 20:1 sowie *brain64* mit 90:1 betrachtet.

Bei *bone64B* reicht eine Optimierung von ca. einem Drittel der Maskenpunkte bereits aus, um den Fehler von $21,11$ auf $18,14$ zu senken und damit den Fehlerwert 1 von JPEG2000 zu unterschreiten. Das Endergebnis die-

ser Optimierung nach EED-Parameteroptimierung und Maskeninversion beträgt $17,8$. Für *brain64* mit dem hohen Kompressionsverhältnis 90:1 sinkt der Fehler durch einen Grauwertoptimierungsschritt von $96,47$ auf $72,62$. Nach der EED-Optimierung und Maskeninversion beträgt der Fehler nur noch $68,42$.

Alle genannten Fehlerwerte sind bis auf die zweite Kommastelle gerundet. Vergleichsbilder für beide Experimente mit und ohne Grauwertoptimierung sind in Abbildung 5.10 enthalten. Beide Bilder wurden mit dem Punktmuster AB erstellt. Für *brain64* wurden weiterhin die Parameter $q = 11$, $l = 1,440773$, $a = 1,913452$ und $\lambda = \frac{1}{158}$ ($\lambda_{\text{opt}} = \frac{1}{151}$ nach der EED-Optimierung) benutzt. Die entsprechenden Parameter für *bone64B* sind $q = 20$, $l = 1,438389$, $a = 0,444336$ und $\lambda = \frac{1}{46}$ ($\lambda_{\text{opt}} = \frac{1}{66}$).

Es ist hierbei festzustellen, dass das Potential der Qualitätsverbesserung durch Grauwertoptimierung in diesen Tests nicht voll ausgeschöpft wurde. In beiden Fällen wurde die Optimierung nicht bis zur Konvergenz weitergeführt, sondern nach einem Optimierungsschritt abgebrochen, da die Qualität von JPEG-2000 bereits nach jeweils einem Schritt übertroffen wurde. Für weitere Schritte sind allerdings weniger signifikante Qualitätsgewinne zu erwarten.

5.3.6 ROI-Codierung

Ein einzelnes Experiment mit ROI-Codierung im R-EED-Verfahren wird im Folgenden diskutiert. Für das Testbild Lenna wurde eine ROI-Maske definiert und der Fehler innerhalb der ROI und des Gesamtbildes einmal mit und einmal ohne ROI-Codierung gemessen.

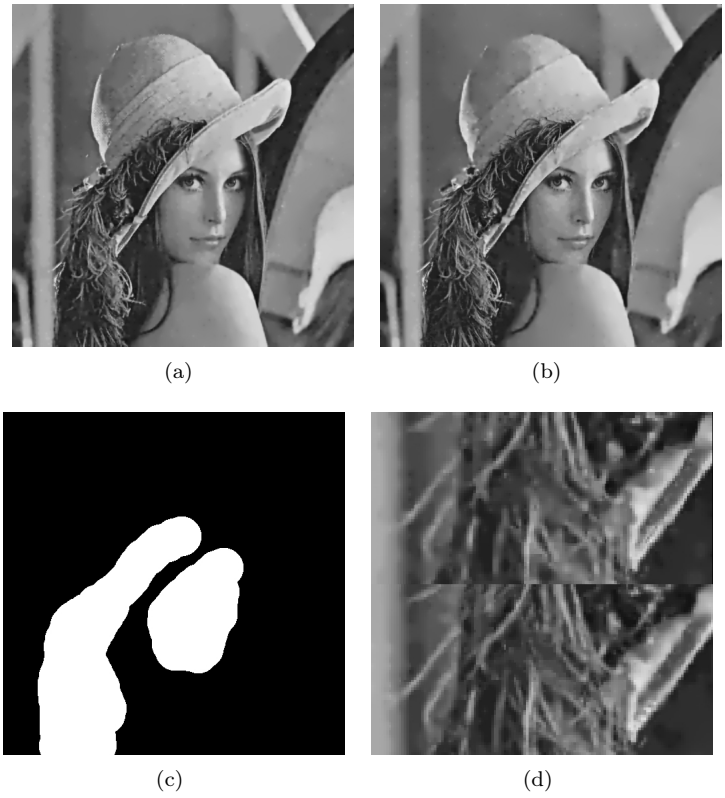
Der MSE in der ROI bei R-EED-Codierung mit Kompressionsverhältnis 20:1 beträgt ohne ROI-Codierung 14,02 (auf die zweite Kommastelle gerundet). Durch ROI-Codierung, die der ROI jeweils die halbe Fehlerschranke des Restbildes zuweist, kann der MSE auf 7,67 verringert werden. Dabei steigt der Gesamtfehler von 39,56 auf 54,73. Im visuellen Vergleich ist ein deutlicher Qualitätsverlust außerhalb der ROI sowie ein Detailgewinn in der ROI zu beobachten (vgl. Abb. 5.11).

Obiges Experiment bestätigt, dass sich R-EED und C-EED aufgrund ihrer Struktur leicht um eine ROI-Funktionalität erweitern lassen. Der gemessene Qualitätsgewinn innerhalb der ROI erfüllt dabei die Erwartungen, allerdings stellt sich für die zukünftige Integration von ROI-Methoden in EED-Verfahren die Frage, ob der Anstieg des Gesamtfehlers verhindert oder abgemildert werden kann.

5.4 Zusammenfassung

Die Experimente ergaben, dass der bei C-EED2D durch eine Erweiterung um adaptive Quaderunterteilung erzielte Speicherplatzgewinn ausreicht, um R-EED auf dreidimensionalen Daten bei fixierter Kompressionsrate qualitativ zu übertreffen. Dieser Gewinn reicht in einigen Fällen sogar schon aus, um mit C-EED2D ohne Grauwertoptimierung eine bessere Qualität als mit JPEG2000 zu erzielen.

Zusätzlich ist durch 3D-Inpainting im C-EED-Verfahren unter Idealbedingungen eine Fehlerminderung von über fünfzig Prozent gegenüber C-EED2D erreichbar. Im Praxistest mit medizinischen Daten reicht dies aus, um JPEG2000



(a) (b)

(c) (d)

Abbildung 5.11: **ROI-Experiment.** Abbildung (a) zeigt das Testbild Lenna nach Kompression im Verhältnis 20:1 mit R-EED. Durch ROI-Codierung (Abb. (b)) mit der ROI-Maske aus Abbildung (c) werden Details in der ROI auf Kosten der Qualität des Restbildes besser erhalten. Insbesondere der Federbusch profitiert von der ROI-Codierung, wie in Abbildung (d) zu sehen ist: der obere Bildausschnitt stammt aus Bild (a), der untere aus (b)

je nach Bildinhalt der Eingabedaten für moderate bis hohe Kompressionsverhältnisse zu übertreffen. Durch Grauwertoptimierung und Maskeninversion kann C-EED zudem in allen betrachteten Stichproben, in denen C-EED zunächst JPEG2000 unterlegen schien, eine höhere Qualität als der waveletbasierte Kompressionsalgorithmus erzielen.

Schwachstellen offenbart C-EED2D auf 3D-Daten mit ähnlicher Abtastrate in allen drei Dimensionen nur für konstruierte synthetische Bilder, die keinen Zusammenhang zwischen aufeinanderfolgenden Schnittbildern aufweisen. Es ist daher anzunehmen, dass der Qualitätsgewinn durch 3D-Inpainting bei niedrigen Abtastraten in der dritten Dimension (z.B. bei Filmausschnitten mit schnellen Bewegungen), abnimmt. Dies wurde zwar durch Experimente mit klassischen Test-Bildsequenzen verifiziert, allerdings sind derartige Bildsequenzen nur begrenzt repräsentativ, da klassische Testbilder oft nur wenige Frames enthalten (weniger als zwanzig). Da C-EED unter solchen Einschränkungen nicht konkurrenzfähig zu Videokompressionsmethoden wie MPEG4 wäre, wurden in dieser Arbeit derartige Tests nicht behandelt. Die Eignung von C-EED für Videokom-

pression bietet sich jedoch als Gegenstand zukünftiger Forschung an (siehe auch Abschnitt 6.2).

Insgesamt erweist sich C-EED als effektiver Kompressionsalgorithmus für dreidimensionale Daten, deren Abtastrate in der z-Dimension nicht signifikant von der in x- und y-Richtung abweicht. Die Möglichkeit, JPEG-2000 allein durch den mit 3D-Inpainting erzielten Qualitätsgewinn zu übertreffen, ohne wie R-EED auf Grauwertoptimierung zurückzugreifen, demonstriert das Potential dieser neuen Methode.

Kapitel 6

Zusammenfassung und Ausblick

6.1 Zusammenfassung

In dieser Arbeit wurde, aufbauend auf dem R-EED-Verfahren von Schmaltz et al. (2010), mit C-EED ein neuer PDE-basierter Kompressionsalgorithmus für dreidimensionale Bilddaten eingeführt. C-EED beruht auf dem gleichen Prinzip wie R-EED: Anhand einer adaptiven Bildunterteilung wird eine Teilmenge der Bildpunkte des Originals derart bestimmt, dass durch Inpainting mit kantenverstärkender Diffusion das Ursprungsbild möglichst exakt rekonstruiert werden kann. Diese Inpaintingmaske wird mittels Quantisierung und Entropie-Codierung effizient gespeichert und durch zusätzliche Kompressionsschritte optimiert.

C-EED behält dieses Grundprinzip bei, ersetzt allerdings die rechteckige Bildunterteilung des R-EED-Verfahrens durch eine dreidimensionale Quaderunterteilung. Auf die resultierende Inpaintingmaske wird 3D-Inpainting angewendet, um ungenutztes Potential der zusätzlichen Dimension für eine Verbesserung der Kompressionsqualität zu erschließen. Zusätzlich ermöglicht die Definition einer dreidimensionalen Inpaintingmaske die effizientere Ausnutzung von Redundanzen durch Entropie-Codierung. Der so gewonnene Speicherplatz kann ebenfalls zu einer Verbesserung der Bildqualität genutzt werden. Um den Qualitätsgewinn durch die dreidimensionale Kompression getrennt von den Verbesserungen durch ganzheitliche Entropie-Codierung zu untersuchen, wurde ein C-EED-Derivat namens C-EED2D eingeführt, welches die dreidimensionale Quaderunterteilung mit zweidimensionalem Inpainting auf x-y-Schnittbildern kombiniert.

In Experimenten mit C-EED und C-EED2D wurden Vergleiche sowohl zu dem zweidimensionalen Vorgänger R-EED als auch zu JPEG2000, dem leistungsfähigsten Kompressionsalgorithmus des DICOM-Standards für medizinische 3D-Bilddaten, gezogen. Bereits C-EED2D ist in der Lage, R-EED und, auf ausgewählten Bildern, auch JPEG2000 zu übertreffen. C-EED wiederum ermöglicht eine signifikante Qualitätssteigerung gegenüber C-EED2D und übertrifft bei Anwendung adäquater Optimierungsmethoden JPEG2000 auf nahezu allen Testbildern.

Zudem wurde in dieser Arbeit demonstriert, dass sich C-EED und R-EED aufgrund der adaptiven Bildunterteilung gut dafür eignen, wichtige Bildteile (ROI) auf Kosten des Restbildes mit höherer Qualität zu komprimieren. Diese Eigenschaft begünstigt zusätzlich die Eignung von C-EED für den praktischen Einsatz bei der Kompression medizinischer Daten.

Insgesamt belegen die Erkenntnisse aus den durchgeführten Experimenten, dass mit C-EED eine Grundlage für die Etablierung konkurrenzfähiger, PDE-basierter Kompressionsmethoden für 3D-Bilddaten geschaffen wurde.

6.2 Ausblick

In den folgenden Abschnitten werden Perspektiven für zukünftige Forschung auf der Basis von C-EED erörtert.

Vergleich mit 3D-Waveletverfahren

In dieser Arbeit wurde das Hauptaugenmerk auf den Vergleich mit R-EED und die Kompression medizinischer Bilddaten gelegt. Da JPEG2000 die effektivste Kompressionsmethode des DICOM-Standard darstellt, stellte dies auch den Maßstab für die Eignung von R-EED dar. Die Experimente wurden zwar so angelegt, dass JPEG2000 in begrenztem Maße Redundanzen in der dritten Dimension ausnutzen konnte (siehe Kapitel 5.2), dennoch bleibt JPEG2000 eine zweidimensionale Kompressionsmethode.

Dreidimensionale, transformationsbasierte Kompressionsmethoden (Baskurt et al., 1995; Kim und Pearlman, 1997, 1999; Secker und Taubman, 2002; Lewis und Knowles, 1990) konnten wegen mangelnder Standardisierung und des zeitlich begrenzten Rahmens der Arbeit nicht mit C-EED verglichen werden. Als dreidimensionales Verfahren sollte C-EED in zukünftigen Untersuchungen jedoch auch mit diesen Methoden verglichen werden.

Numerische Verfahren

Die Implementierung des PDE-basierten Inpaintings beruht in dieser Arbeit auf SOR mit sukzessiver Verfeinerung. Wie in Kapitel 4.2 dargelegt, wurde SOR aufgrund der guten Laufzeit als numerisches Lösungsverfahren für das Gleichungssystem des EED-Inpaintings gewählt. Allerdings stellt die fehlende Rotationsinvarianz einen Nachteil dar.

Neben der Untersuchung der Eignung alternativer iterativer Lösungsverfahren für die implizite Methode bietet sich mit dem FED-Verfahren von Grewenig et al. (2010) seit kurzem eine Alternative auf der Basis eines expliziten Diffusionsansatzes. FED erlaubt es, Restriktionen für die Zeitschrittweite in expliziten Verfahren zu umgehen und wurde bei zweidimensionaler Bildkompression mit EED-Inpainting bereits erfolgreich eingesetzt. Aufgrund der Rotationsinvarianz und der unkomplizierteren Implementierung dieser Methode bietet sich eine Untersuchung der Eignung von FED für den Einsatz im C-EED-Verfahren an.

ROI-Codierung

In dieser Arbeit wurde ROI-Codierung in C-EED und R-EED lediglich in einer Machbarkeitsstudie betrachtet, da die dreidimensionale Erweiterung der PDE-

basierten Kompressionsmethode den Hauptgegenstand der Arbeit darstellt. Die ROI-Funktionalität wurde daher lediglich behandelt, um die Eignung zur Kompression medizinischer Daten zu betonen.

In zukünftigen Arbeiten kann dieser Ansatz weiter verfolgt und ausführlich untersucht werden, inwiefern sich die Qualität in der ROI verbessern lässt und die gleichzeitige Zunahme des MSE für das Gesamtbild vermieden werden kann. Im Idealfall sollte zwar die Qualität außerhalb der ROI abnehmen, die Gesamtqualität des Bildes jedoch möglichst wenig sinken.

Nach eingehender Untersuchung dieser Sachverhalte und einer Verbesserung der ROI-Codierung in C-EED und R-EED kann ein Vergleich zu der ROI-Funktionalität von Transformationscodierern wie JPEG2000 erfolgen.

Videokompression

Die durchgeführten Experimente zeigen, dass sich C-EED grundsätzlich für Videocodierung eignet, wenngleich auch die Laufzeit der gegenwärtigen Implementierung das Codieren von Videos in praxistauglichen Auflösungen verhindert. Köstler et al. (2007) haben jedoch gezeigt, dass sich mittels Parallelisierung auf entsprechender Hardware Videos mit R-EED in Echtzeit komprimieren lassen, was ähnliche Möglichkeiten für C-EED nahelegt. Zudem lag der Fokus dieser Arbeit darauf, möglichst optimale Resultate zu erzielen. In der Praxis können schnellere Parameteroptimierungsmethoden, die der optimalen Qualität lediglich ausreichend nahe kommen, bereits genügen, um befriedigende Ergebnisse zu erzielen.

Neben der Laufzeitkomponente ist insbesondere die Behandlung der Zeitdimension ein wichtiges Element der Anwendung von C-EED auf Videos. Bei räumlichen Daten aus bildgebenden Verfahren sind die Maßeinheiten aller drei Dimensionen identisch, auch wenn nicht in jeder Dimension bei der Abtastung die gleiche Gitterweite eingesetzt wird. In der Zeitdimension ist es dagegen nichttrivial, eine ideale Gitterweite zu finden, da das Größenverhältnis zwischen Zeiteinheit und Raumeinheit nicht fest definiert ist.

Insbesondere für niedrige Abtastraten in der Zeitdimension gestaltet es sich als schwierig, Redundanzen in Videoaufnahmen mit sich schnell bewegenden Objekten, Kamerabewegungen o.ä. durch 3D-Inpainting auszunutzen. In zukünftigen Arbeiten könnte untersucht werden, ob nach dem Vorbild von Bewegungskompensation (siehe Strutz, 2002) Strategien entwickelt werden können, welche die Qualität von C-EED verbessern.

Da Videos in der Regel viele künstliche Schnitte enthalten, muss auch untersucht werden, wie eine Videosequenz in Teilsequenzen zerlegt werden kann, die eine möglichst optimale Kompression mit C-EED erlauben. Eine derartige Zerlegung ist auch unter dem Gesichtspunkt der Laufzeit von Bedeutung, damit Dekompression in adäquater Geschwindigkeit erfolgen kann.

Eine Erweiterung des C-EED-Verfahrens auf Video-Codierung würde das Anwendungspotential des Algorithmus signifikant vergrößern und den Weg für PDE-basierte Videokompression bereiten.

Literaturverzeichnis

Alter, F, Durand, S, und Froment, J. Adapted total variation for artifact free decompression of JPEG images. *Journal of Mathematical Imaging and Vision*, 23(2):199–211. 2005.

Bae, E und Weickert, J. Partial differential equations for interpolation and compression of surfaces. In Daehlen, M, Floater, M, Lyche, T, Merrien, J.-L, Mørken, K, und Schumaker, L. L (Hrsg.), *Mathematical Methods for Curves and Surfaces*, Lecture Notes in Computer Science. Springer, Berlin, 2010.

Bajaj, C, Ihm, I, und Park, S. 3D RGB image compression for interactive applications. *ACM Transactions on Graphics*, 20:10–38. 2001.

Barrett, R, Berry, M, Chan, T, Demmel, J, Donato, J, Dongarra, J, Eijkhout, V, Pozo, R, Romine, C, und Van der Vorst, H. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2. Auflage, 1994.

Baskurt, A, Benoit-Cattin, H, und Odet, C. 3D medical image coding method using a separable 3D wavelet transform. In Kim, Y (Hrsg.), *Medical Imaging 1995: Image Display*, Nr. 2431 von *Proceedings of SPIE*, S. 173–183. SPIE Press, Bellingham, USA, 1995.

Battiato, S, Gallo, G, und Stanco, F. Smart interpolation by anisotropic diffusion. In *Proc. Twelvth International Conference on Image Analysis and Processing*, S. 572–577, Montova, Italy, 2003. IEEE Computer Society Press.

Belhachmi, Z, Bucur, D, Burgeth, B, und Weickert, J. How to choose interpolation data in images. *SIAM Journal on Applied Mathematics*, 70(1):333–352. 2009.

Bertalmío, M, Bertozzi, A, und Sapiro, G. Navier–Stokes, fluid dynamics, and image and video inpainting. In *Proc. 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Nr. 1, S. 355–362, Kauai, HI, 2001. IEEE Computer Society Press.

Bertalmío, M, Sapiro, G, Caselles, V, und Ballester, C. Image inpainting. In *Proc. SIGGRAPH 2000*, S. 417–424, New Orleans, LI, 2000.

Black, M. J und Anandan, P. The robust estimation of multiple motions: parametric and piecewise smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104. 1996.

Bornemann, F und März, T. Fast image inpainting based on coherence transport. *Journal of Mathematical Imaging and Vision*, 28(3):259–278. 2007.

Bourdon, P, Augereau, B, Chatellier, C, und Olivier, C. MPEG-4 compression artifacts removal on color video sequences using 3D nonlinear diffusion. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, Nr. 3, S. 729–732, Montreal, Canada, 2004. IEEE Computer Society Press.

Bourne, R. *Fundamentals of Digital Imaging in Medicine*. Springer, London, 2010.

Bradley, A und Stentiford, F. JPEG 2000 and region of interest coding. In *Proc. Digital Image Computing Techniques and Applications*, Nr. 2, S. 303–308, 2002.

Briggs, W. L. *A Multigrid Tutorial*. SIAM, Philadelphia, 1987.

Burrows, M und Wheeler, D. A block-sorting lossless data compression algorithm, 1994. Digital SRC Research Report.

Carslaw, H. S und Jaeger, J. C. *Conduction of Heat in Solids*. Oxford University Press, Oxford, 2. Auflage, 1959.

Caselles, V, Morel, J.-M, und Sbert, C. An axiomatic approach to image interpolation. *IEEE Transactions on Image Processing*, 7(3):376–386. 1998.

Chan, T. F und Zhou, H. M. Feature preserving lossy image compression using nonlinear PDE's. In Luk, F. T (Hrsg.), *Advanced Signal Processing Algorithms, Architectures, and Implementations VIII*, Nr. 3461 von *Proceedings of SPIE*, S. 316–327. SPIE Press, Bellingham, USA, 1998.

Chan, T und Shen, J. Inpainting based on nonlinear transport and diffusion. In Nashed, M. Z und Scherzer, O (Hrsg.), *Inverse Problems, Image Analysis, and Medical Imaging*, Nr. 313 von *Contemporary Mathematics*, S. 53–65. American Mathematical Society, Providence, USA, 2002a.

Chan, T und Shen, J. Mathematical Models for local nontexture inpaintings. *SIAM Journal on Applied Mathematics*, 62(3):1019–1043. 2002b.

Chan, T und Shen, J. Variational image inpainting. *Communications on Pure and Applied Mathematics*, 58(5):579–619. 2005.

Crank, J. *The Mathematics of Diffusion*. Oxford University Press, Oxford, 2. Auflage, 1975.

Distasi, R, Nappi, M, und Vitulano, S. Image compression by B-tree triangular coding. *IEEE Transactions on Communications*, 45(9):1095–1100. 1997.

Esedoglu, S und Shen, J. Digital Inpainting Based on the Mumford–Shah–Euler Image Model. *European Journal of Applied Mathematics*, 13(04):353–370. 2002.

Ford, G. E. Application of inhomogeneous diffusion to image and video coding. In *Proc. 13th Asilomar Conference on Signals, Systems and Computers*, Nr. 2, S. 926–930, Asilomar, CA, 1996. IEEE Computer Society Press.

Ford, G. E, Estes, R. R, und Chen, H. Scale-space analysis for image sampling and interpolation. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Nr. 3, S. 165–168, San Francisco, CA, 1992.

Galić, I, Weickert, J, Welk, M, Bruhn, A, Belyaev, A, und Seidel, H.-P. Towards PDE-based image compression. In Paragios, N, Faugeras, O, Chan, T, und Schnörr, C (Hrsg.), *Variational, Geometric and Level-Set Methods in Computer Vision*, Nr. 3752 von *Lecture Notes in Computer Science*, S. 37–48. Springer, Berlin, 2005.

Galić, I, Weickert, J, Welk, M, Bruhn, A, Belyaev, A, und Seidel, H.-P. Image compression with anisotropic diffusion. *Journal of Mathematical Imaging and Vision*, 31(2–3):255–269. 2008.

Gokturk, S, Tomasi, C, Girod, B, und Beaulieu, C. Medical image compression based on region of interest, with application to colon CT images. In *Proc. 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Nr. 3, S. 2453–2456, Istanbul, Turkey, 2001. Engineering in Medicine and Biology Society (EMES).

Gothandaraman, A, Whitaker, R, und Gregor, J. Total variation for the removal of blocking effects in DCT based encoding. In *Proc. 2001 IEEE International Conference on Image Processing*, Nr. 2, S. 455–458, Thessaloniki, Greece, 2001.

Gourlay, A. R. Hopscotch: a fast second-order partial differential equation solver. *IMA Journal of Applied Mathematics*, 6(4):375–390. 1970.

Grewenig, S, Weickert, J, und Bruhn, A. From box filtering to fast explicit diffusion. In Goesele, M, Roth, S, Kuijper, A, Schiele, B, und Schindler, K (Hrsg.), *Lecture Notes in Computer Science*, Nr. 6376, S. 533–542. Springer, Berlin, 2010.

Huffman, D. A. A method for the construction of minimum redundancy codes. *Proceedings of the IRE*, 40:1098–1101. 1952.

Karlson, G und Vetterli, M. Three-dimensional sub-band coding of video. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, S. 1100–1103, New York, 1988. IEEE Computer Society Press.

Keefer, E, Pfeifer-Schäller, I, Sauerwein, C, und Wälischmiller, H. Voxel and STL-Data in Service of Archaeology - Digital Celts. In *Proc. European Conference on Non-Destructive Testing*, Berlin, 2006. The Open Access NDT Database. URL http://www.ndt.net/article/ecndt2006/papers~1.htm.

Ketcham, P und Feder, D. Visualizing Bose-Einstein condensates. *Computing in Science and Engineering*, 5(1):86–89. 2003.

Kim, B und Pearlman, W. An embedded wavelet video coder using three-dimensional set partitioning in hierarchical trees (SPIHT). In *Proc. Data Compression Conference*, S. 251–260, Los Alamitos, CA, 1997. IEEE Computer Society Press.

Kim, Y und Pearlman, W. Lossless volumetric medical image compression. In Tescher, A. G (Hrsg.), *Applications of Digital Image Processing XXII*, Proceedings of SPIE, S. 305–312. SPIE Press, San Diego, USA, 1999.

Kopilovic, I und Szirányi, T. Artifact reduction with diffusion preprocessing for image compression. *Optical Engineering*, 44(2):1–14. 2005.

Köstler, H, Stürmer, M, Freundl, C, und Rüde, U. PDE based video compression in real time. Technical Report 07-11, Lehrstuhl für Informatik 10, Univ. Erlangen–Nürnberg, Germany, 2007.

Lehmann, T, Gönner, C, und Spitzer, K. Survey: Interpolation methods in medical image processing. *IEEE Transactions on Medical Imaging*, 18(11): 1049–1075. 1999.

León, L, Lam, D, Schertzer, W, und Swayne, D. Lake and climate models linkage: a 3D hydrodynamic contribution. *Advances in Geoscience*, 4:57–62. 2005.

Lewis, A und Knowles, G. Video compression using 3D wavelet transforms. *Electronics Letters*, 26(6):396–398. 1990.

Mahoney, M. Adaptive weighing of context models for lossless data compression. Technical Report CS-2005-16, Florida Institute of Technology, Melbourne, Florida, 2005.

Mahoney, M. Data compression programs. http://mattmahoney.net/dc/, 2010. Letzte Überprüfung 10. Januar 2011.

Mainberger, M und Weickert, J. Edge-based image compression with homogeneous diffusion. In Jiang, X und Petkov, N (Hrsg.), *Computer Analysis of Images and Patterns*, Nr. 5702 von *Lecture Notes in Computer Science*, S. 476–483. Springer, Berlin, 2009.

Martin, G. Range encoding: an algorithm for removing redundancy from a digitised message. In *Proc. International Conference on Video and Data Recording*, Southampton, England, 1979. Institution of Electronic and Radio Engineers.

Masnou, S. Disocclusion: a variational approach using level lines. *IEEE Transactions on Image Processing*, 11(2):68–76. 2002.

Masnou, S und Morel, J.-M. Level lines based disocclusion. In *Proc. IEEE International Conference on Image Processing*, Nr. 3, S. 259–263, Chicago, IL, 1998. IEEE Computer Society Press.

Meister, A. *Numerik linearer Gleichungssysteme*. Vieweg, Braunschweig, 1999.

Mémin, E und Pérez, P. Hierarchical estimation and segmentation of dense motion fields. *International Journal of Computer Vision*, 46(2):129–155. 2002.

Moving Picture Experts Group. Information Technology – Coding of Audio-Visual Objects – Part 2: Visual. ISO/IEC JTC 1 14496-2, 2004.

Moving Picture Experts Group. Information Technology – Coding of Audio-Visual Objects – Part 10: Advanced Video Coding. ISO/IEC JTC 1 14496-10, 2009.

National Electrical Manufacturers Association. Digital Imaging and Communications in Medicine (DICOM) – Part 5 Data Structures and Encoding. PS 3.5-2004, 2004.

Novo, A, Grasmueck, M, Viggiano, D, und Lorenzo, H. 3D GPR in archaeology: what can be gained from dense data acquisition and processing? In *Proc. Twelfth International Conference on Ground Penetrating Radar*, S. 16–19, Birmingham, United Kingdom, 2008. University of Birmingham Press.

OsiriX. OsiriX DICOM Sample Image Sets, 2009. URL http://pubimage.hcuge.ch:8080/. Letzte Überprüfung: 10. Januar 2011.

Pennebaker, W. B und Mitchell, J. L. *JPEG: Still Image Data Compression Standard*. Springer, New York, 1992.

Perona, P und Malik, J. Scale space and edge detection using anisotropic diffusion. In *Proc. IEEE Computer Society Workshop on Computer Vision*, S. 16–22, Miami Beach, FL, 1987. IEEE Computer Society Press.

Rosset, A, Spadola, L, und Ratib, O. OsiriX: An Open-Source Software for Navigating in Multidimensional DICOM Images. *Journal of Digital Imaging*, 17(3):205–216. 2004.

Roussos, A und Maragos, P. Vector-valued image interpolation by an anisotropic diffusion-projection PDE. In Sgallari, F, Murli, F, und Paragios, N (Hrsg.), *Scale Space and Variational Methods in Computer Vision*, Nr. 4485 von *Lecture Notes in Computer Science*, S. 104–115. Springer, Berlin, 2007.

Schmaltz, C, Weickert, J, und Bruhn, A. Beating the quality of JPEG 2000 with anisotropic diffusion. In Denzler, J, Notni, G, und Süße, H (Hrsg.), *Pattern Recognition*, Nr. 5748 von *Lecture Notes in Computer Science*, S. 452–461. Springer, Berlin, 2009.

Schmaltz, C, Weickert, J, und Bruhn, A. Advances in image compression with anisotropic diffusion. unveröffentlicht., 2010.

Secker, A und Taubman, D. Highly scalable video compression using a lifting-based 3D wavelet transform with deformable mesh motion compensation. In *Proc. IEEE International Conference on Image Processing*, Nr. 3, S. 749–752, Rochester, USA, 2002. IEEE Computer Society Press.

Strobach, P. Quadtree-structured recursive plane decomposition coding of images. *IEEE Transactions on Signal Processing*, 39(6):1380–1397. 1991.

Strutz, T. *Bilddatenkompression*. Vieweg, Braunschweig, 2002.

Sullivan, G. J und Baker, R. J. Efficient quadtree coding of images and video. *IEEE Transactions on Image Processing*, 3(3):327–331. 1994.

Taubman, D. S und Marcellin, M. W (Hrsg.). *JPEG 2000: Image Compression Fundamentals, Standards and Practice*. Kluwer, Boston, 2002.

Telea, A. An image inpainting technique based on the fast marching method. *Journal of Graphics Tools*, 9(1):23–34. 2004.

Tönnies, K. *Grundlagen der Bildverarbeitung*. Pearson Studium, München, 2005.

Tschumperlé, D. Fast anisotropic smoothing of multi-valued images using curvature-preserving PDE's. *International Journal of Computer Vision*, 68 (1):65–82. 2006.

Tschumperlé, D und Deriche, R. Vector-valued image regularization with PDE's: a common framework for different applications. In *Proc. 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Nr. 1, S. 651–656, Madison, WI, 2003. IEEE Computer Society Press.

Tsuji, H, Sakatani, T, Yashima, Y, und Kobayashi, N. A nonlinear spatio-temporal diffusion and its application to prefiltering in MPEG-4 video coding. In *Proc. IEEE International Conference on Image Processing*, Nr. 1, S. 85–88, Rochester, NY, 2002. IEEE Computer Society Press.

Tsuji, H, Tokumasu, S, Takahashi, H, und Nakajima, M. Spatial prefiltering scheme based on anisotropic diffusion in low-bitrate video coding. *Systems and Computers in Japan*, 38(10):34–45. 2007.

Varga, R. A. *Matrix Iterative Analysis*. Prentice Hall, Englewood Cliffs, 1962.

Weickert, J. *Anisotropic Diffusion in Image Processing*. PhD thesis, Department of Mathematics, University of Kaiserslautern, Germany, 1996a. Revised and extended version published by Teubner, Stuttgart, Germany, 1998.

Weickert, J. Theoretical foundations of anisotropic diffusion in image processing. *Computing Supplement*, 11:221–236. 1996b.

Weickert, J. Nonlinear diffusion filtering. In Jähne, B, Haußecker, H, und Geißler, P (Hrsg.), *Handbook on Computer Vision and Applications, Vol. 2: Signal Processing and Pattern Recognition*, S. 423–450. Academic Press, San Diego, 1999.

Weickert, J. Diskretisierung anisotroper Diffusionsprozesse, 2010. private Korrespondenz.

Weickert, J und Welk, M. Tensor field interpolation with PDEs. In Weickert, J und Hagen, H (Hrsg.), *Visualization and Processing of Tensor Fields*, S. 315–325. Springer, Berlin, 2006.

Welch, T. A. A technique for high-performance data compression. *Computer*, 17(6):8–19. 1984.

Yang, S und Hu, Y. Coding artifact removal using biased anisotropic diffusion. In *Proc. 1997 IEEE International Conference on Image Processing*, Nr. 2, S. 346–349, Santa Barbara, CA, 1997. IEEE Computer Society Press.

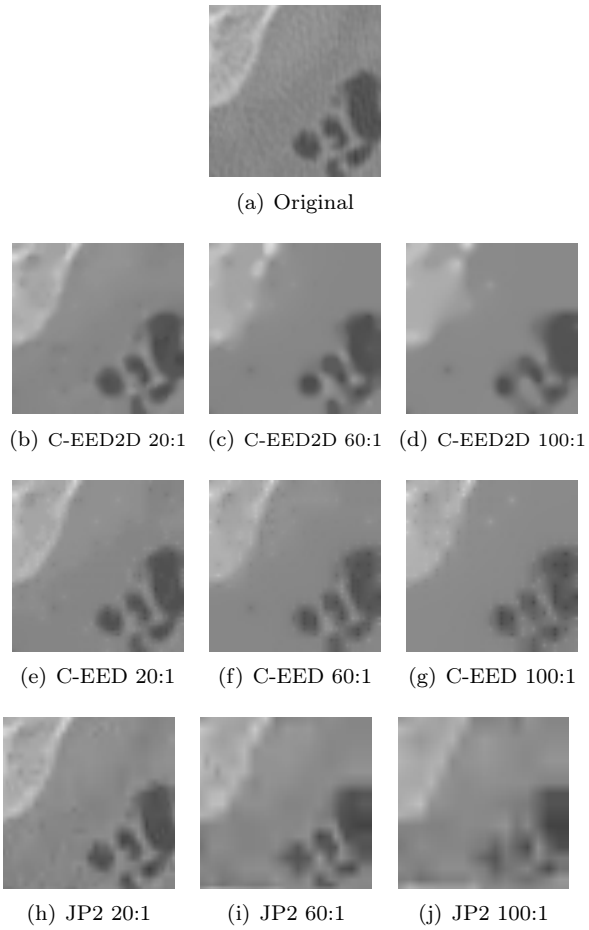
Ziv, J und Lempel, A. An universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3):337. 1977.

Abbildungsverzeichnis

Anhang A

Zusätzliche Kompressionsergebnisse

Auf den folgenden Seiten werden zusätzliche Kompressionsergebnisse von C-EED, C-EED2D und JPEG2000 dargestellt. Die Abbildungen A.1 und A.2 enthalten Gegenüberstellung von Schnittbildern aus *bone64B* und *transition* mit Kompressionsraten 20:1, 60:1 und 100:1.



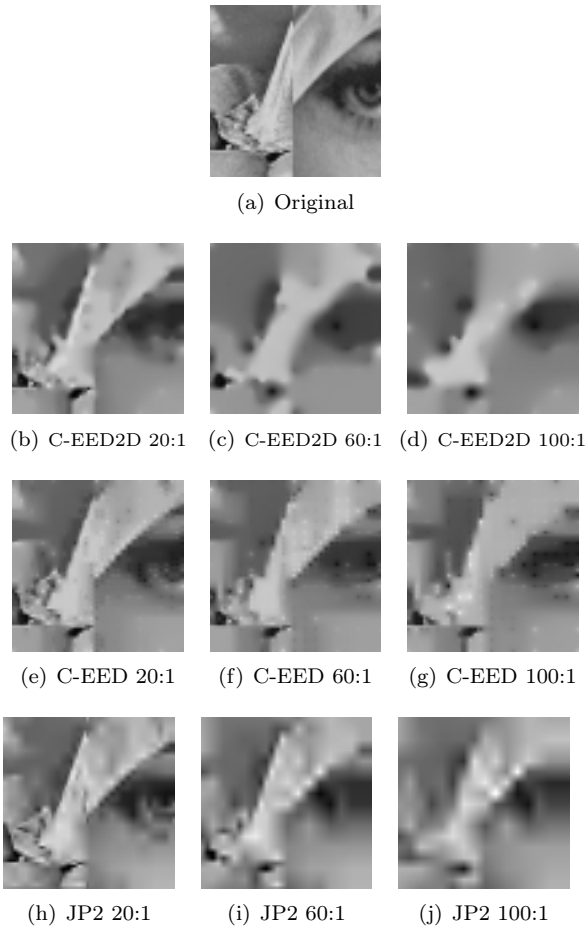
(a) Original

(b) C-EED2D 20:1 (c) C-EED2D 60:1 (d) C-EED2D 100:1

(e) C-EED 20:1 (f) C-EED 60:1 (g) C-EED 100:1

(h) JP2 20:1 (i) JP2 60:1 (j) JP2 100:1

Abbildung A.1: **Visueller Vergleich: bone64B.** Die obigen Abbildungen zeigen Kompressionsergebnisse der Algorithmen C-EED, C-EED2D und JPEG2000 (Abgekürzt zu JP2) angewendet auf das vierte Schnittbild von *bone64B*. C-EED und C-EED2D nutzen jeweils das Muster AB, Parameteroptimierung von λ, l, a und q. Grauwertoptimierung und Maskeninversion wurden nicht durchgeführt.



(a) Original

(b) C-EED2D 20:1 (c) C-EED2D 60:1 (d) C-EED2D 100:1

(e) C-EED 20:1 (f) C-EED 60:1 (g) C-EED 100:1

(h) JP2 20:1 (i) JP2 60:1 (j) JP2 100:1

Abbildung A.2: **Visueller Vergleich: transition.** Die obigen Abbildungen zeigen Kompressionsergebnisse der Algorithmen C-EED, C-EED2D und JPEG2000 (Abgekürzt zu JP2) angewendet auf das zweiunddreißigste Schnittbild von *transition*. C-EED und C-EED2D nutzen jeweils das Muster AB, Parameteroptimierung von λ, l, a und q. Grauwertoptimierung und Maskeninversion wurden nicht durchgeführt.

Anhang B

Bedienung der Referenzimplementierung

Die Referenzimplementierung enthält nach dem Vorbild der R-EED-Implementierung von Schmaltz et al. (2010) zwei Teilprogramme, die verschiedene Arten der Parameteroptimierung durchführen können. Im Folgenden wird die Bedienung der beiden Programme *cm* und *optimize* erläutert.

Kompression mit cm

Das Programm *cm* enthält grundlegende Funktionen zur C-EED- und C-EED2D-Kompression mit und ohne Parameteroptimierung. Ein Programmaufruf erfolgt nach dem Schema *./cm -i <Eingabedatei> -c <komprimierte Datei> -o <Ausgabedatei> [optionale Parameter]*, wobei folgende Parameter erlaubt sind:

-a <x> Setzt die adaptive Fehlerschranke für die Quaderunterteilung auf den Wert x.

-d <x> Wählt den Diffusionstyp x für den Inpaintingschritt (0: explizites Verfahren, 1: FED (experimentell), 2: SOR, 3: 2D-Schnittbild-SOR).

-e <x> Setzt den Kontrastparameter λ der kantenverstärkenden Diffusion auf x.

-f <x> Wählt die Art der Entropie-Codierung aus (0: Keine Entropie-Codierung, 1: Huffman, 3: statische Bereichscodierung 4: Bereichscodierung 5: PAQ, 6: gzip, 7: bzip2).

-H <x> Setzt den Hopscotch-Parameter auf x.

-l <x> Setzt die Stufenanpassung l der Quaderunterteilung auf den Wert x.

-L <x> Setzt den Glättungsparameter σ auf x.

-M <x> Setzt die minimale Baumtiefe auf x.

-N <x> Setzt die maximale Baumtiefe auf x.

-o <x> Setzt den Dateinamen der Ausgabedatei im pdm-Format.

-O <**x**> Optimiert die Parameter x, wobei sich x als Zeichenketten aus den Optionen a: adaptive Fehlerschranke, e: Kontrastparameter, p: Punktmuster, q: Grauwertanzahl, l: Stufenanpassung zusammensetzt. Die Optimierungen werden in der angegebenen Reihenfolge durchgeführt. Empfehlenswert ist aus Laufzeitgründen, e und q als erste Glieder der Zeichenkette zu wählen.

-p <**x**> Setzt Punktmuster auf x (0: B, 1: D, 2: C, 3: A, 4: F, 5: E, 6: AB, 7: AD, 8: AC, 9: AF, 10: AE, 11: BC, 12: BD, 13: BE, 14: BF).

-P <**x**> Aktiviert Bildpartitionierung mit Tiefe x. Das Bild wird gleichmäßig in dreidimensionale Teilbilder mit Tiefe x zerlegt, die einzeln komprimiert werden. Die Parameterwahl -P 1 entspricht einer R-EED-Kompression der Schnittbilder.

-q <**x**> Setzt die Anzahl der Grauwerte für die Quanitisierung auf x.

-S <**x**> Bestimmt das minimale Kompressionsverhältnis x:1.

-z <**x**> Bestimmt die Skalierung der x- und y-Dimension relativ zur z-Dimension.

Optimierung mit optimize

Das Programm *optimize* operiert auf bereits im C-EED-Format gespeicherten, komprimierten Dateien und lässt nachträgliche Änderungen und Parameteroptimierungen zu. Die primären Funktionen sind Grauwertoptimierung, Interpolationsumkehr sowie Auswahl der optimalen Entropie-Codierung.

Ein Aufruf von *optimize* entspricht dem Schema: *./optimize -i <Eingabedatei> -c <komprimierte Datei> -o <Ausgabe-Datei> [optionale Parameter]*, wobei die folgenden Parameter erlaubt sind:

-C Setzt den Optimierungsvorgang bis zur Konvergenz fort, indem optimize nach jedem Optimierungsschritt erneut aufgerufen wird, bis keine weitere Verbesserung mehr erzielt werden kann.

-e Bewirkt die Optimierung des EED-Kontrastparameter λ.

-E Speichert eine Visualisierung des Fehlers in dem Bild error.pdm.

-f Sucht die optimale Art der Entropie-Codierung.

-F <**x**> Setzt die Entropie-Codierung Nummer x ein (0: Keine Entropie-Codierung, 1: Huffman, 3: statische Bereichscodierung 4: Bereichscodierung 5: PAQ, 6: gzip, 7: bzip2).

-h Bewirkt Optimierung des Hopscotch-Parameters (Interpolationsumkehr).

-H <**x**> Setzt den Hopscotch-Parameter auf den Wert x.

-q <**x**> Führt Grauwertoptimierung mit Quantisierungsparameter x durch.

-s <**x**> Startet Grauwertoptimierung mit dem x-ten Punkt.