

Institut für Visualisierung und Interaktive Systeme

Universität Stuttgart  
Universitätsstraße 38  
D-70569 Stuttgart

Bachelorarbeit Nr. 77

# **HDR-Bildfusion mit gleichzeitiger Schätzung der Kamera-Antwortkurve**

Sebastian Zillessen

**Studiengang:** Softwaretechnik  
**Prüfer:** Prof. Dr.-Ing. Andrés Bruhn  
**Betreuer:** Prof. Dr.-Ing. Andrés Bruhn

**Beginn am:** 20. Juni 2013  
**Beendet am:** 19. Dezember 2013

**CR-Nummer:** G.1.2, G.1.3, G.1.6, G.1.8, I.3.3, I.3.6,  
I.4.0, I.4.1, I.4.3, I.4.8, I.4.9



## Kurzfassung

In den letzten Jahren hat die Bedeutung von High Dynamic Range (HDR) Bildern im aufstrebenden Forschungsgebiet *Computational Photography* stark zugenommen. HDR-Bilder können große Helligkeitsunterschiede detailreich wiedergeben.

Das Verhältnis zwischen hellstem und dunkelstem Bildpunkt wird als Dynamikumfang bezeichnet. Bilder mit erweitertem Dynamikumfang können durch die Fusionierung von Einzelaufnahmen einer Belichtungsserie auch ohne spezielle Hardware erzeugt werden. Das in dieser Ausarbeitung verwendete Verfahren von Debevec und Malik schätzt neben der Berechnung des HDR-Bildes auch die kameraspezifische Antwortkurve des Bildaufnahmeprinzips mithilfe einer Energiefunktion.

Die vorliegende Arbeit stellt einen alternierenden Lösungsansatz für dieses Verfahren vor, durch das die Verwendung aller Bildpunkte der Belichtungsserie bei der Berechnung des HDR-Bildes möglich ist. Darüber hinaus werden drei Erweiterungen der Energiefunktion vorgestellt: Das Einführen einer Forderung von Monotonie der Antwortkurve soll die physikalische und radiometrische Korrektheit verbessern. Die Berechnung des HDR-Bildes wird um einen (räumlichen) Glattheitsterm erweitert, der insbesondere bei Rauschen zu einer verbesserten Ausgabe führen soll. Zudem werden die quadratischen Bestrafungsterme des Ausgangsverfahrens durch subquadratische Funktionen ersetzt. Dies soll zu einer Verbesserung des Verfahrens bezüglich der Robustheit gegenüber Fehlmessungen und Ausreißern führen.

Neben der theoretischen Ausarbeitung der Erweiterungen des Verfahrens wird darüber hinaus eine Realisierung in Java vorgestellt. Mithilfe dieser Software werden die Einflüsse der verschiedenen Erweiterungen experimentell untersucht und evaluiert.

## Abstract

High Dynamic Range (HDR) image technology has gained ever more importance in the thriving field of *Computational Photography*. HDR images allow for highly detailed display of lighting contrasts.

In this context, the ration between the brightest and darkest pixels is referred to as dynamic range. It is possible to compose images with highly increased dynamic range independent of dedicated hardware by merging single pictures of a series.

This dissertation discusses the method developed by Debevec and Malik which estimates the device specific response curve using an energy function in order to calculate the HDR image. The author then offers an alternating method, which allows to calculate a HDR image over all pixels of an image series. Furthermore, three additions to the discussed energy formula will be proposed: Introducing a monotony condition to the response curve improves the physical and radiometric accuracy. Adding a (spatial) smoothness term to the calculation of the radiance map improves output, especially when handling noise in the input. Finally, replacing the quadric penalty term in the algorithm with a subquadric function which will improve robustness against measurement errors and outliers.

In addition to the theoretical discussion on improving the energy formula, a Java implementation is presented. Using this software, the impact of the proposed additions were experimentally tested and evaluated.

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>7</b>
1.1	Motivation . . . . .	7
1.2	Aufgabenstellung . . . . .	8
1.3	Gliederung . . . . .	8
<b>2</b>	<b>Grundlagen der HDR-Bilder</b>	<b>11</b>
2.1	Prinzip . . . . .	12
2.2	Anwendungsgebiet und Geschichte . . . . .	13
2.3	Bilderzeugung . . . . .	14
2.4	Bildformate und -speicherung . . . . .	15
2.5	Bilddarstellung . . . . .	16
2.6	Software zur Erstellung von HDR-Bildern . . . . .	18
<b>3</b>	<b>Verwandte Arbeiten und Implementierungen</b>	<b>21</b>
3.1	Bekannte Implementierungen des Ansatzes von Debevec und Malik . . . . .	21
3.2	Verwandte Arbeiten . . . . .	21
<b>4</b>	<b>Algorithmus von Debevec und Malik</b>	<b>23</b>
4.1	Ansatz . . . . .	23
4.2	Berechnung der Antwortkurve . . . . .	25
4.3	Konstruktion der Radiance Map . . . . .	25
4.4	Mögliche Erweiterungen des Ansatzes . . . . .	26
<b>5</b>	<b>Mathematische Ausarbeitung</b>	<b>29</b>
5.1	Optimierungsansatz . . . . .	29
5.2	Erweiterung um Monotonie-Eigenschaft . . . . .	33
5.3	Erweiterung um einen räumlichen Glattheitsterm . . . . .	35
5.4	Erweiterung um Robustheit . . . . .	37
5.5	Lösung der Gleichungssysteme . . . . .	43
<b>6</b>	<b>Realisierung</b>	<b>45</b>
6.1	Anforderungen . . . . .	45
6.2	Wahl der Programmiersprache . . . . .	47
6.3	Prototyping . . . . .	48
6.4	Architektur . . . . .	50
6.5	Externe Bibliotheken . . . . .	54
6.6	Programmvorstellung . . . . .	55

<b>7</b>	<b>Ergebnisse und Resultate</b>	<b>59</b>
7.1	Ergebnisse mit Monotonie-Bedingung . . . . .	60
7.2	Ergebnisse mit räumlichem Glattheitsterm . . . . .	60
7.3	Ergebnisse mit subquadratischen Bestrafungstermen . . . . .	61
<b>8</b>	<b>Zusammenfassung und Ausblick</b>	<b>67</b>
<b>A</b>	<b>Anhang</b>	<b>69</b>
A.1	LU-Zerlegung . . . . .	69
A.2	MTB-Algorithmus . . . . .	70
	<b>Glossar und Abkürzungsverzeichnis</b>	<b>73</b>
	<b>Literaturverzeichnis</b>	<b>75</b>

# 1 Einleitung

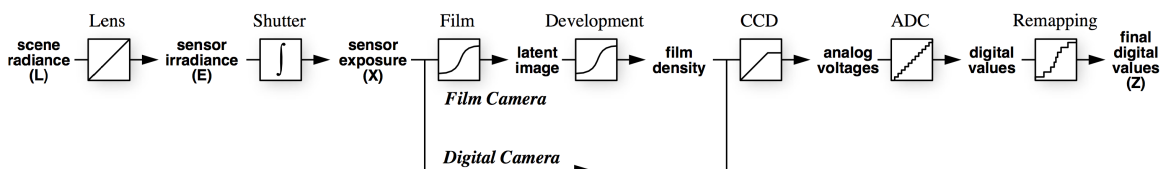
Die High Dynamic Range (HDR) Bildgebung ist eines von vielen interessanten Problemen in dem aufstrebenden Forschungsgebiet *Computational Photography*. Bei diesen Bildern handelt es sich um Repräsentationen einer realen Szene, die im Vergleich zu herkömmlichen Aufnahmen einen erhöhten Dynamikumfang besitzen und somit ein sehr hohes Verhältnis zwischen hellsten und dunkelsten Bildpunkt aufweisen können. Ziel dieser Arbeit ist, die Fusion mehrerer Bilder mit verschiedener Belichtungszeit, zu einem einzigen Bild mit deutlich vergrößertem Dynamikumfang.

## 1.1 Motivation

Während sich viele Arbeiten nur mit der pixelweisen Fusion der Bilddaten auseinander setzen, schlagen Debevec und Malik [DM97] vor, gleichzeitig auch noch die Antwortkurve des Bildaufnahmeprozesses, d.h. der verwendeten Kamera, mitzuschätzen. Diese Antwortkurve ist die kameraspezifische Abbildung, welche aus den Beleuchtungswerten der aufzunehmenden Szene Grau- bzw. Farbwerte erzeugt (siehe Abbildung 1.1).

Dies bietet den klaren Vorteil, die Bildfusion auch ohne vorherige radiometrische Kalibration des Aufnahmeequipments durchführen zu können. Als mathematisches Werkzeug zur Formulierung des Verfahrens dient hierbei ein gemeinsames Energiefunktional, das einen Ähnlichkeits- und einen Glattheitsterm besitzt. Während der Ähnlichkeitsterm unter Berücksichtigung der mit geschätzten Antwortkurve die Beziehung zwischen den Einzelaufnahmen und dem gesuchten HDR-Bild herstellt, sorgt der Glattheitsterm für eine hinreichend glatte Antwortkurve, die auch aus radiometrischer Sicht Sinn ergibt.

Trotz der allgemeinen Formulierung hat das Verfahren von Debevec und Malik jedoch auch einige Schwachstellen. Zum einen werden weder im Daten- noch im Glattheitsterm

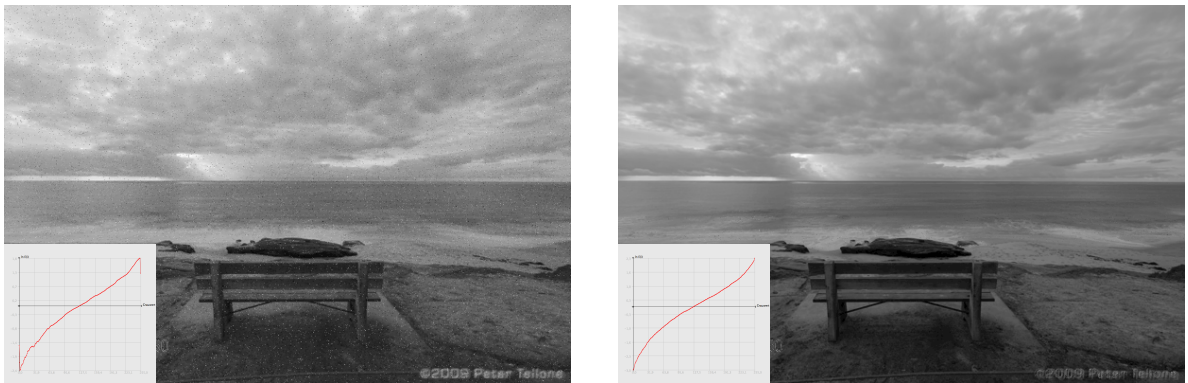


**Abbildung 1.1:** Bildaufnahme-Pipeline — Veranschaulichung der zu durchlaufenden Prozesse bei der Aufnahme eines Bildes mit einer Kamera [DM97, S.2].

robuste Bestrafungsfunktionen verwendet. Diese könnten den Ansatz deutlich robuster unter Fehlmessungen machen. Zum anderen werden keine Beschränkungen gefordert, die die typischerweise gewünschte Monotonie der Antwortkurve explizit erzwingen. Monotone Kurven können deshalb nur bei einer hinreichend großen Gewichtung der Glattheit erzielt werden. Schließlich ist das Verfahren auch nicht sonderlich robust gegenüber Rauschen, was insbesondere bei sehr kurz belichteten Bildern Probleme bereiten kann.

### 1.2 Aufgabenstellung

Ziel der Arbeit ist es zunächst das Verfahren von Debevec und Malik [DM97] als Ausgangsverfahren in einer (dafür sinnvollen und portierbaren) Programmiersprache zu implementieren. Diese Implementierung soll anschließend sukzessive um eine Monotonie-Beschränkung (siehe Abschnitt 5.2), einen räumlichen Glattheitsterm (siehe Abschnitt 5.3) und robuste Bestrafungsfunktionen (siehe Abschnitt 5.4) erweitert werden.



**Abbildung 1.2:** *Antwortkurven (inkl. zugehörigem HDR-Bild)* — Das HDR-Bild wurde mittels lokalem Reinhard-Tone-Mapper (siehe Abschnitt 2.5) erstellt. **links:** Ohne robuste Bestrafungsterme. **rechts:** Mit robusten Bestrafungstermen und räumlicher Glattheitsforderung.

Neben der Modellierung und Implementierung der einzelnen Erweiterungen soll auch eine geeignete visuelle Evaluation der Ergebnisse erfolgen. Hierzu sollen Tone-Mapping-Verfahren (siehe Abschnitt 2.5) aus bereits existierender Forschung verwendet werden.

### 1.3 Gliederung

In Kapitel 2 werden zunächst die Grundlagen der HDR-Bilder vermittelt. Dabei wird auf die physikalischen, historischen und anwendungsorientierten Eigenschaften von HDR-Bildern eingegangen. Im anschließenden Kapitel 3 werden verwandte Arbeiten zum Thema HDR



zusammengefasst sowie existierende Implementierungen des Ansatzes von Debevec und Malik vorgestellt.

Die zu Grunde liegende Vorgehensweise von Debevec und Malik wird im nachfolgenden Kapitel 4 beschrieben. Darüber hinaus werden die existierenden Schwachstellen des bisherigen Ansatzes dargestellt und erläutert. In Kapitel 5 folgt die mathematische Ausarbeitung des Ansatzes sowie die Formulierung der Erweiterungen des Algorithmus.

Im Kapitel 6 wird die Herangehensweise an die Problemstellung aus Sicht der Software-Entwicklung beschrieben. Es werden Anforderungen an das zu implementierende Programm formuliert, ein Prototyp vorgestellt und die Architektur der Software beschrieben. Abschließend wird die Verwendung der Anwendung erläutert.

Das Kapitel 7 stellt die Resultate und Einflüsse der unterschiedlichen Erweiterungen vor. Die Ergebnisse werden mit dem Standard-Ansatz verglichen und es folgt eine Analyse der Resultate. Das Kapitel 8 rundet die vorliegende Arbeit mit einer Zusammenfassung der Ergebnisse und Ausblicken ab.



## 2 Grundlagen der HDR-Bilder

In den vergangenen Jahren hat die digitale Fotografie zu einem Umdenken und einer Neuschaffung von Kommunikationskanälen geführt. Durch die Verbreitung von Digitalkameras (insbesondere auch solchen, die in Smartphones und Handys eingebaut sind) können Nachrichten in Form von Fotografien in Sekundenschnelle über den ganzen Globus geschickt werden. In Diensten wie Twitter, Instagram oder Vine werden Bilder und Videos in großen Mengen verschickt. Nach aktuellen Zahlen werden z.B. in Instagram täglich 55 Millionen Bilder gepostet<sup>1</sup>. Damit spielt auch die digitale Bildverarbeitung eine immer größer werdende Rolle.

Digitalbilder werden in der heutigen Zeit hauptsächlich in Form der drei Farbkanäle Rot, Grün und Blau dargestellt (sog. RGB-Farbraum). Häufig kommt noch ein vierter Kanal, der sog. Alpha-Kanal hinzu, der für die Darstellung von Transparenz genutzt wird.

Jeder der Kanäle wird in der Regel mittels eines Bytes repräsentiert. Damit können 16.7 Millionen verschieden Farben dargestellt werden. Trotz dieser großen Zahl sind nur 256 verschiedene Werte für jeden Farbkanal möglich. Diese Anzahl ist häufig unzureichend, um Szenen mit hohen Helligkeitsunterschieden darzustellen [RWPD10, S. 1f].



**Abbildung 2.1:** HDR-Bild mit bereits angewandtem Tone-Mapping-Verfahren [Tel10]

<sup>1</sup><http://instagram.com/press/>

Die Verwendung von HDR-Bildern kann diese Problematik beheben. Ziel ist es, mehr Farben und Details in unterschiedlichen Bildbereichen sichtbar zu machen. Um dies zu ermöglichen, erhöht man bei HDR-Bildern den Dynamikumfang des Bildbereiches. Dazu ist jedoch eine andere Form der Repräsentation des Bildes notwendig (sog. Radiance Maps).

### 2.1 Prinzip

Das menschliche Auge kann in einer täglichen Szene einen Dynamikumfang im Bereich von 1 : 10.000 wahrnehmen [FJ04]. Dieser Umfang liegt weit über den herkömmlichen Werten eines normalen Kamera-Sensors. In der Tabelle 2.1 sind verschiedene Dynamikumfänge (und die damit zusammenhängenden Belichtungsstärken) aufgelistet. Um mit HDR-Bildern einen höheren Dynamikumfang darstellen zu können, müssen mehr Informationen aus der Szene ausgelesen werden, als durch herkömmliche Verfahren möglich ist. Dazu werden entweder mehrere Bilder mit verschiedenen Belichtungszeiten zu einer Radiance Map kombiniert oder es werden spezielle Kamera-Sensoren eingesetzt, welche in der Lage sind eine höhere Dichte an Bildinformationen (z.B. große Helligkeitsunterschiede) aufzunehmen [YGFT99].

Der hier verwendete Begriff „Dynamikumfang“ beschreibt das Verhältnis zwischen dem hellsten und dunkelsten Pixel im Bild. Um Ausreißern einen geringeren Einfluss zu geben und die Messung robuster zu machen, werden manchmal auch Quantile verwendet, wodurch Rauschen in der Eingabe weniger stark gewichtet wird. Bei Bildschirmen wird hingegen unter dem Dynamikumfang das Verhältnis zwischen der maximalen und minimalen Leuchtkraft verstanden [RWPD10, S. 4].

Nach der Vorstellung des Prinzips von HDR-Bildern folgt nun eine kurze Auflistung von möglichen Anwendungsgebieten.

Umgebung	Belichtungsstärke ( $cd/m^2$ )
Sternenhimmel	$10^{-3}$
Mondschein	$10^{-1}$
Innenraum Beleuchtung	$10^2$
Sonnenlicht	$10^5$
Herkömmliche Monitore	$10^2$

**Tabelle 2.1:** Belichtungsstärken in verschiedenen Umgebungen [RWPD10, S. 6]

## 2.2 Anwendungsgebiet und Geschichte

Die Möglichkeiten des Einsatzes von HDR-Bildern sind vielfältig. Die nachfolgende Liste umfasst einige der Gebiete, in denen diese Technologie eingesetzt wird oder werden kann [RWPD<sub>10</sub>, S. 87f].

**Digitale Fotografie:** Die verschiedenen Kamera-Hersteller tendieren zu sog. „aufnahme-abhängigen Daten“. In diesen sind häufig mehr Bildinformationen enthalten. Dabei handelt es sich bei verschiedenen Herstellern in der Regel um verschiedene Rohdatenformate von Kameraaufnahmen (RAW), die meist untereinander nicht kompatibel sind.

**Satellitenbilder:** Satellitenbilder beinhalten in aller Regel sehr viel mehr Informationen als nur den sichtbaren Bereich des Lichtspektrums. HDR-Bilder sind hier von Bedeutung, da sie multispektrale Aufnahmen ermöglichen.

**Visualisierungen und Rendering:** HDR-Bilder wurden vermutlich erstmals in Render-Engines von Visualisierungen (Computer-Spiele, medizinische Visualisierungen und Simulationen, etc.) eingesetzt. Bei manchen Anwendungen ist es insbesondere für Reflektionen wichtig, auch nicht sichtbare Frequenzen bei Berechnungen mit einzubeziehen, da diese durch Interferenzen wieder sichtbar werden können und somit der Detailgrad steigt.

**Bildbearbeitungssoftware:** Die großen Bildbearbeitungsanwendungen bieten mittlerweile in der Regel ebenfalls die Bearbeitung und Generierung von HDR-Bildern an (z.B. Adobe Photoshop<sup>2</sup>, Photogenics<sup>3</sup> und Photomatix<sup>4</sup>).

**Medizin:** In der Endoskopie besteht ein großer Bedarf an hoch auflösenden Complementary Metal Oxide Semiconductor (CMOS) Bildsensoren. Diese können immer bessere Aufnahmen aus dem Inneren des Körpers liefern und ermöglichen der Medizin große Fortschritte. Solche Sensoren können bereits in der Größe eines Streichholzkopfes ein Kontrastverhältnis von mehr als  $1 : 10^7$  (179 dB) erreichen [BGH<sup>+</sup>06].

**Virtual Reality:** In Anwendungen, bei denen sich der Benutzer in einem virtuellen Raum bewegt, wird die Wahrnehmung zunehmend wichtig. Auch hier spielen deshalb hohe Dynamikumfang eine besondere Rolle. In diesem Bereich ist es besonders wichtig, gute Kompressions-Algorithmen für HDR-Bilder zu entwickeln, um eine schnelle Übertragung zu gewährleisten. Auch beim Platzieren von synthetischen Objekten in realen Szenen können HDR-Bilder eingesetzt werden, um dem Betrachter eine noch „realere“ Szene zu suggerieren [Deb98].

<sup>2</sup><http://adobe.com/photoshop>

<sup>3</sup><http://www.cinepaint.org>

<sup>4</sup><http://www.hdrsoft.com/download.html>

Um HDR-Bilder in diesen Bereichen verwenden zu können, müssen diese zunächst erzeugt werden. Im nachfolgenden Abschnitt werden drei verschiedene Möglichkeiten dazu vorgestellt.

### 2.3 Bilderzeugung

Für die Erstellung von HDR-Bildern gibt es unterschiedliche Möglichkeiten. Dabei muss jedoch zwischen echten HDR-Bildern und Pseudo-HDR-Bildern unterschieden werden. Im Nachfolgenden werden die verschiedenen Verfahren kurz beschrieben. Der Fokus liegt dieser Arbeit liegt auf dem letzten Verfahren, der HDR-Bildgenerierung aus einer Belichtungsreihe, da dieses auch im Ausgangsverfahren von Debevec und Malik verwendet wird.

#### 2.3.1 Pseudo-HDR-Bilder

Bei Pseudo-HDR-Bildern handelt es sich um eine einfache Fusion von Bildreihen. Deswegen werden diese Verfahren auch Exposure Blending oder Exposure Fusion genannt. Bei dieser Technologie geht es darum, mehr Details aus einer Belichtungsreihe von LDR-Bildern (Low Dynamic Range Bilder) zu generieren, ohne dabei ein HDR-Bild zu erzeugen. Die Bilder der Belichtungsreihe werden dazu einfach fusioniert [LZR12].

#### 2.3.2 HDR-Kameras

Diese speziellen Kameras verfügen über Bildsensoren, die von sich aus einen höheren Dynamikumfang aufnehmen können und dadurch die notwendigen Informationen in einer Aufnahme generieren können. Diese Spezial-Kameras sind jedoch noch sehr teuer und wenig verbreitet [Blo12, S. 95ff]. Digitale Spiegelreflex-Kameras bieten mittlerweile häufig ebenfalls einen HDR-Modus an.

#### 2.3.3 HDR-Bildgenerierung aus einer Belichtungsreihe

Um ein HDR-Bild aus einer Belichtungsreihe zu erzeugen, braucht man zunächst eine Grundlage für das Bild. In diesem Verfahren werden mehrere Bilder der selben Szene mit unterschiedlichen Belichtungszeiten fusioniert. Ziel der dabei verwendeten Algorithmen ist es, aus diesen Bildern ein HDR-Bild zu erzeugen.

Um die Bilder später weiter zu verarbeiten, müssen diese jedoch zunächst registriert werden. Dieser Prozess beschreibt die gegenseitige Ausrichtung der Bilder einer Belichtungsreihe, so dass diese genau übereinander liegen. Dadurch werden Verschiebungen in den Belichtungsreihen eliminiert, die z.B. durch das Bewegen der Kamera entstanden sind. Eine solche Registrierung ist aufgrund der verschiedenen Belichtungswerte der Aufnahmen häufig nicht

oder nur schlecht über Kantendetektions-Verfahren möglich, da diese Merkmale unter den unterschiedlichen Belichtungen sehr stark variieren können.

Ein besserer Ansatz um Bilder zu registrieren ist der Mean Threshold Bitmap Alignment (MTB) Ansatz (siehe Abschnitt A.2), da dieser ohne Kantendetektions-Verfahren auskommt. In den nachfolgenden Vergleichen und in der Implementierung wurde auf die Implementierung eines solchen Verfahrens verzichtet, da nur vorab registrierte Bilder verwendet werden.

Die so erstellten HDR-Bildern sollen nun auch gespeichert werden können. Das nachfolgende Kapitel beschäftigt sich deshalb mit der Kodierung, Komprimierung und Abspeicherung von HDR-Bildern.

## 2.4 Bildformate und -speicherung

Für die Abspeicherung der HDR-Bilder werden in Tabelle 2.2 die drei gängigsten Formate mit den gängigen dazugehörigen Kodierungen verglichen [RWPD<sub>10</sub>, S. 89]. Die Speicherung der erzeugten Daten war kein zentraler Bestandteil der vorliegenden Arbeit und wurde deshalb bei der Realisierung nicht berücksichtigt. Dennoch sollen hier die bekanntesten Kodierungen zur Abspeicherung vorgestellt werden.

Format	Kodierung	Kompression	Metadaten	Lizenz
HDR	RGBE	Lauflängen- kodierung	Kalibrierung, Farbraum	Open source software ( <i>Radiance</i> )
	XYZE	Lauflängen- kodierung	+ benutzerdef. Daten	
TIFF	IEEE RGB	keine	Kalibrierung, Farbraum	Public domain library ( <i>libtiff</i> )
	LogLuv24	keine	+ Registrierung + benutzerdef. Daten	
	LogLuv32	Lauflängen- kodierung		
EXR	Half RGB	Wavelet, ZIP	Kalibrierung, Farbraum + Fensterfunktion + benutzerdef. Daten	Open source library ( <i>OpenEXR</i> )

**Tabelle 2.2:** *HDR-Bildformate* — Die verbreitetsten Formate und ihre Eigenschaften in der Übersicht [RWPD<sub>10</sub>, S.89].

### 2.4.1 RGBE – Das .hdr Format

Dieses Datei-Format wurde ursprünglich unter den Dateierendungen `.hdr` und `.pic` eingeführt. Abgesehen von den Metadaten (wie z.B. Bildgröße, Ausrichtung, notwendige Angaben zur verwendeten Kodierung, etc.) wird jeder Bildpunkte mit 32-Bit dargestellt, die sich auf die Kanäle für Rot, Grün und Blau sowie einen Exponenten verteilen. Die Darstellung der Kanäle zusammen mit dem Exponenten führt zu einer Vergrößerung des Dynamikbereiches [RWPD<sub>10</sub>, S. 92].

### 2.4.2 TIFF – Gleitkomma Kodierung

Das Format `.tiff` enthält eine 32-Bit Kodierung pro Komponente (also 96-Bit für einen Bildpunkt). Dabei werden die Bildpunkte mittels Fließkommazahlen dargestellt [Ado92]. Dieser Standard unterstützt bereits eine sehr hohe Genauigkeit. Dazu benötigt dieses Dateiformat im Vergleich zu anderen jedoch am meisten Speicherplatz. Allerdings lässt sich damit auch eine nahezu verlustfreie Abspeicherung von HDR-Bildern erreichen. Im Standard von 1992 wurde auf jede Form der Komprimierung verzichtet [RWPD<sub>10</sub>, S. 93]. Dieser kann jedoch um verschiedene Kompressionsverfahren erweitert werden. LogLuv ist beispielsweise ein solches, bei dem die Werte logarithmisch skaliert und quantisiert werden [Lar98].

### 2.4.3 EXR – EXTended Range Format

Dieses Format wurde 2002 veröffentlicht<sup>5</sup> und basiert ebenfalls auf der Speicherung von Fließkommazahlen. Es existiert eine Variante bei der nur 16 Bit (Hälfte der normalen Anzahl) für das Speichern der Fließkommazahlen verwendet wird: ein Bit für das Vorzeichen, fünf für den Exponenten und zehn für die Mantisse. Für diese Komprimierung sind Quantisierungsschritte von unter 0.1% vorgesehen und damit für das menschliche Auge nicht erkennbar. Dadurch ist diese Kompression quasi verlustfrei durchführbar [RWPD<sub>10</sub>, S. 97f].

## 2.5 Bilddarstellung

Das Erstellen von HDR-Bildern (siehe Abschnitt 2.3) erzeugt sog. Radiance Maps. Dabei handelt es sich um eine Zuordnung der Bestrahlungsstärke durch das einfallende Licht zu jedem Bildpunkt einer Szene. Für die Darstellung dieser Radiance Maps gibt es vereinzelt spezielle Hardware, die in der Lage ist den erweiterten Dynamikumfang darzustellen. Sehr viel häufiger kommen jedoch sog. Tone-Mapping (dt.: Dynamikkompressions) Verfahren zum Einsatz. Diese stellen ein HDR-Bild durch eine andere Skalierung des Bildbereichs als herkömmliche Bilddateien dar.

<sup>5</sup>[www.openexr.com](http://www.openexr.com)



Der Kerngedanke beim Tone-Mapping besteht darin, einen geeigneten Weg für die Zuordnung von Bildpunkten aus dem HDR-Bild in das Low Dynamic Range (LDR) Bild zu finden [Blo12, S. 145]. Diese Zuordnungsfunktionen nennen sich Tone-Mapping-Operatoren und können generell in zwei Kategorien unterschieden werden. Die globalen Operatoren (siehe Unterabschnitt 2.5.1) bearbeiten alle Bildpunkte gleich, während die lokalen Operatoren (siehe Unterabschnitt 2.5.2) Informationen aus der Umgebung in die Berechnung an jedem Bildpunkt mit einbeziehen.

### 2.5.1 Globale Tone-Mapping-Operatoren

Bei globalen Tone-Mapping-Operatoren wird die gesamte Farbkurve (engl. tone curves) modifiziert. Bei dieser Zuordnung handelt es sich um eine Korrelation zwischen Eingabe (HDR-Bild) und Ausgabe (LDR-Bild), die die Umrechnung der Belichtungsstärke in Farbwerte angibt. Die Veränderungen durch den Tone-Mapping-Operator können auf den verschiedenen Farbkanälen unterschiedlich sein. Auch die Berechnung der modifizierten Kurve kann sich aufgrund des Bildes ändern [Blo12, S. 146].

Es bestehen viele verschiedene Ansätze für globale Tone-Mapping-Operatoren, die unterschiedliche Vor- und Nachteile aufweisen. In dieser Arbeit wurde lediglich der Tone-Mapping-Operator von Reinhard et al. [RSSFo2] implementiert und verwendet. Dabei handelt es sich um die vereinfachte Form des komplexeren lokalen Operators, der im gleichen Artikel veröffentlicht wurde. Hierzu wird zunächst der durchschnittliche Wert des Logarithmus aus der Helligkeit des Bildes  $L_w(x, y)$  bestimmt. Dieser wird als charakteristischer Wert der Szene  $\tilde{L}_w$  beschrieben. Anschließend werden die skalierten Helligkeiten des Bildes errechnet (siehe Gleichung 2.1). Der Parameter  $\alpha$  bestimmt die Lage des mittleren Grauwertes und hat in der Regel den Wert 0.18. Daraus lässt sich dann der globale Operator in Gleichung 2.2 erstellen.

$$L(x, y) = \frac{\alpha}{\tilde{L}_w} L_w(x, y) \quad (2.1)$$

$$L_d(x, y) = \frac{L(x, y)}{1 + L(x, y)} \quad (2.2)$$

### 2.5.2 Lokale Tone-Mapping-Operatoren

Lokale Tone-Mapping-Operatoren können bei der Zuordnung eines Wertes aus einem HDR- in ein LDR-Bild auch die lokale Umgebung eines jeden Bildpunktes berücksichtigen. Damit erreichen sie in sehr dynamischen Bildern bessere Ergebnisse und können mehr Details in Bildern hervorheben. Auch hier gibt es eine Vielzahl verschiedener Operatoren, die alle ihre Vor- und Nachteile haben. Da der Operator von Reinhard et al. [RSSFo2] in verschiedenen Veröffentlichungen [YBMS05, KYL<sup>+</sup>07] gut abgeschnitten hat, wird dieser in der vorliegenden Arbeit verwendet.

Bei diesem handelt es sich um einen lokalen Tone-Mapping-Operator, der dodging-and-burning (dt. abwedeln) zur Berechnung verwendet. Dieses Verfahren ist eine Technik, die aus

der analogen Fotografie stammt. Dabei wird die Belichtungszeit in einzelnen Bereichen des Bildes verändert, um diese bei der Entwicklung des Filmmaterials differenziert zu behandeln. Die Wahl der einzelnen Bereiche wird im technischen Ansatz durch die Berechnung des lokalen Kontrastes im Bild umgesetzt, welche die Reichweite des Operators beeinflussen. Auf eine weitere Beschreibung des Verfahrens wird hier aus Gründen der Komplexität verzichtet.

Für den gesamten Prozess der Erstellung, Speicherung und Darstellung von HDR-Bildern gibt es verschiedene Software, welche die einzelnen Schritte durchführen kann. Der folgende Abschnitt stellt einige dieser Anwendungen vor.

### 2.6 Software zur Erstellung von HDR-Bildern

Herkömmliche Programme zur Bildbearbeitung (z.B. Photoshop<sup>6</sup> oder GIMP<sup>7</sup>) unterstützen die Erzeugung von HDR-Bildern aus einer Belichtungsserie recht gut. Es gibt in der Regel mehrere Tone-Mapping-Operatoren, deren Parameter anschaulich verändert werden können. Diese trumpfen mit hohem Funktionsumfang, vielfältigen Einstellungsvarianten und der Möglichkeit der weiteren Bearbeitung auf.

Darüber hinaus gibt es verschiedene Programme, die speziell auf die Erzeugung von HDR-Bildern spezialisiert sind (z.B. Photomatix<sup>8</sup> oder Luminance HDR<sup>9</sup>). Der Funktionsumfang dieser Programme ist verhältnismäßig klein, führt auch Laien schnell zum Ziel, da (wie z.B. bei Luminance HDR, siehe Abbildung 2.2) interaktive Assistenten den Benutzer bei der Erstellung anleiten.

Als weitere Beispiele für HDR-Software seien hier außerdem Dynamic-Photo HDR<sup>10</sup> und HDR Darkroom<sup>11</sup> genannt. Diese Programme verfügen über ein großes Portfolio von Tone-Mapping-Operatoren und können sowohl realistische als auch sehr verfremdete HDR-Bilder generieren.

Ein wirklich einfaches Programm ist Picturenaut<sup>12</sup>. Die Anzahl und der Funktionsumfang der implementierten Tone-Mapping-Operatoren ist limitiert, jedoch liefert das Programm recht rasch realitätsgetreue Bilder.

<sup>6</sup><http://adobe.com/photoshop>

<sup>7</sup><http://www.gimp.org> mit Plugin Exposure Blend ([http://tir.astro.utoledo.edu/jdsmith/code/exposure\\_blend.php](http://tir.astro.utoledo.edu/jdsmith/code/exposure_blend.php))

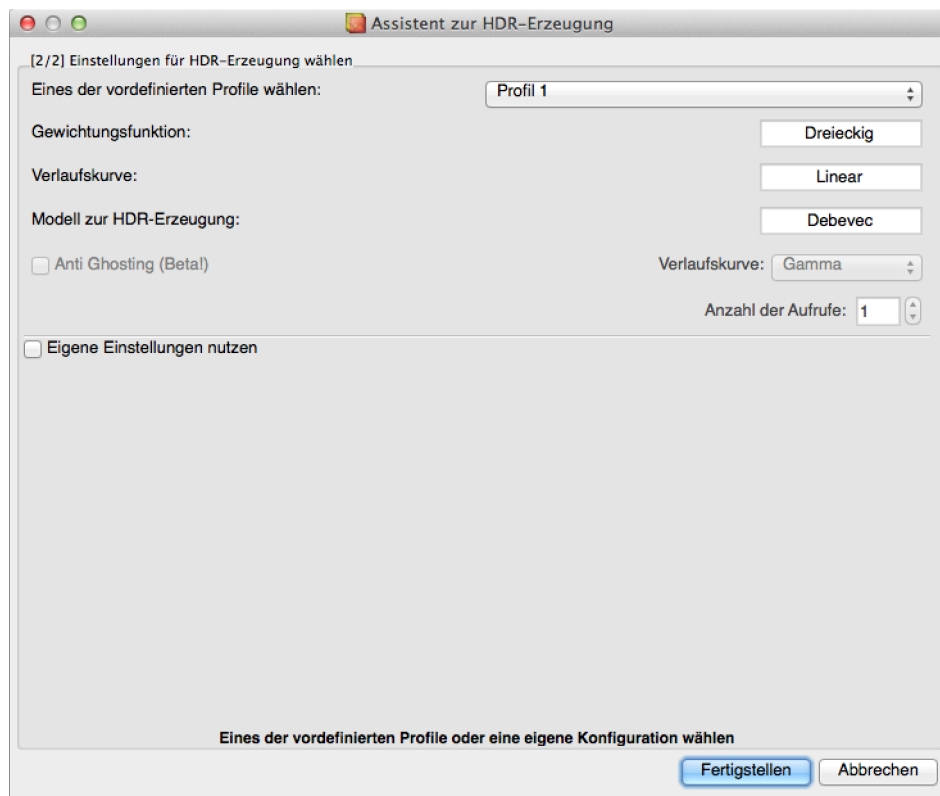
<sup>8</sup><http://www.hdrsoft.com/de/>, kostenpflichtig

<sup>9</sup><http://qtpfsgui.sourceforge.net>, Freeware

<sup>10</sup><http://www.mediachance.com/hdri/index.html>, kostenpflichtig

<sup>11</sup><http://www.everimaging.com>, kostenpflichtig

<sup>12</sup><http://www.hdrlabs.com/picturenaut/>, Freeware



**Abbildung 2.2:** *Erstell-Assistent von Luminance HDR* — Auch hier wird intern der Algorithmus von Debevec und Malik verwendet.



## 3 Verwandte Arbeiten und Implementierungen

In diesem Kapitel soll kurz auf verwandte Arbeiten und Implementierungen eingegangen werden. Der Fokus liegt bei der nachfolgenden Recherche auf dem Ansatz aus einer Belichtungsserie von LDR-Bildern ein HDR-Bild zu generieren.

### 3.1 Bekannte Implementierungen des Ansatzes von Debevec und Malik

Das Standard-Verfahren als solches wurde bereits in verschiedenen Programmen implementiert. Auch der ursprüngliche Artikel von Debevec und Malik veröffentlicht eine MATLAB-Version des Algorithmus. Darauf basierend hat z.B. Mathias Eitz eine Implementierung<sup>1</sup> des kompletten Prozesses in MATLAB geschrieben. Dieser arbeitet ohne Erweiterungen und implementiert direkt den beschriebenen Ansatz. Die Selektion der Bildpunkte, welche bei der Berechnung berücksichtigt werden, geschieht hier jedoch in einer Art Rasterung der Bilder und berücksichtigt damit keine der Forderungen von Debevec und Malik (siehe Unterabschnitt 4.4.2).

Auch in der bereits genannten Software zur Erstellung von HDR-Bildern (siehe Abschnitt 2.6) wird z.T. der Ansatz von Debevec und Malik verwendet.

### 3.2 Verwandte Arbeiten

In den letzten Jahren haben die veröffentlichten Arbeiten zur Generierung, Darstellung und Verarbeitung von HDR-Bildern stetig zugenommen. Mit der nachfolgenden Auswahl von verwandten Arbeiten soll ein Überblick über den Themenbereich geschaffen werden.

Jinno and Okkuda [JO12] beschreiben in ihrer Alternative für die Fusion von Belichtungsserien (basierend auf herkömmlichen Algorithmen) auch die Problematik von sich bewegenden Objekten. Daraus entstehen bei der Fusion häufig ghosting artifacts (dt. Geist-Artefakte) oder motion blur (dt. Bewegungsunschärfe). In dieser Veröffentlichung werden die bewegten Objekte erkannt und bei der Berechnung des HDR-Bildes wieder entfernt. Das Verfahren sagt

<sup>1</sup><http://cybertron.cg.tu-berlin.de/eitz/hdr/index.html>

dabei Überdeckung, Sättigung und Verschiebungen in den Ausgangsbildern voraus und konstruiert die HDR-Bilder unter Berücksichtigung dieser Daten. Damit können insbesondere in Serien mit hoher Bewegung bessere Ergebnisse erzielt werden.

Nayar et al. [NM00] stellen in ihrem Verfahren einen anderen Ansatz der Generierung von HDR-Bildern vor. Dabei wird bereits bei der Aufnahme eines Bildes eine Rasterung durch ein optisches Gitter mit unterschiedlichen Transparenzen erzielt. Das so aufgenommene Bild wird als *spatially varying exposure* (dt. ortsabhängig belichtetes) Bild bezeichnet. Da die Struktur des Gitters und dessen Transparenzen bekannt sind, kann aus dem aufgenommenen Bild nun ein HDR-Bild mit höherem Dynamikumfang berechnet werden. Die unterschiedlichen Transparenzen des Gitters sorgen dafür, dass sowohl hohe als auch niedrige Belichtungen wahrgenommen werden können.

Liu et al. [LG03] beschreiben in ihrem Artikel ein heuristisches Verfahren zur Schätzung der Bewegung in einer Bildserie. Dieses Verfahren basiert auf einem in selbigem Artikel veröffentlichten rekursiven Verfahren bei dem große Belichtungsserien (ihr Beispiel umfasst 65 Aufnahmen) Stück für Stück zu einem HDR-Bild zusammengesetzt werden. Ihr Ansatz verspricht besonders bei Bildern mit sehr schnellen Änderungen (wie z.B. einem sich drehenden Propeller) gute Ergebnisse. Die Aufnahmen selbst werden dabei durch herkömmliche CMOS-Bildsensoren aufgenommen. Mögliche Messfehler werden im Algorithmus ausgiebig diskutiert um Rauschen zu reduzieren.

Im Artikel von Zimmer et al. [ZBW11] wird ebenfalls ein Verfahren zur Reduktion von Bewegungsartefakten bei der Erzeugung von HDR-Bildern beschrieben. Hierbei kommt die Berechnung des optischen Flusses bei der Registrierung der Bilder zum Einsatz. Darüber hinaus wird in diesem Verfahren ein ebenfalls räumlich hochauflösendes HDR-Bild erzeugt, da aus den verschiedenen Bildern der Belichtungsserie durch die Registrierung auch Zwischenpixel-Bereiche mit Informationen gefüllt werden können. Dadurch kann die Auflösung erhöht werden. Mithilfe dieses Verfahrens ist es möglich auch verwackelte Bilder, die Bewegung enthalten, zu registrieren und diese für die HDR-Bild-Generierung zu verwenden.

Auch die Tone-Mapping-Operatoren werden ständig untersucht und verbessert. So vergleichen Kuang et al. [KYL<sup>+</sup>07] in ihrer Studie über HDR-Bildgenerierungs-Algorithmen vier lokale und zwei globale Operatoren miteinander. Während der Studie werden verschiedene Bildszenen mit den sechs Operatoren von Probanden in drei unterschiedlichen Experimenten bewertet. Dazu werden die Bilder paarweise auf einem LDR-Bildschirm gezeigt. Das Experiment stellte fest, dass keiner der untersuchten Operatoren durchweg in allen Szenen besser als alle anderen Probanden abschneidet. Dies führt zur Schlussfolgerung, dass eine große Korrelation zwischen Szene und Tone-Mapping-Verfahren besteht.

In einer Arbeit von Yoshida et al. [YBMS05] werden sieben Tone-Mapping-Operatoren im Direktvergleich der realen Szene und dem korrespondierendem LDR-Bild von Testpersonen bewertet. Berücksichtigt wurden dabei sowohl globale als auch lokale Operatoren. Eine der Hauptkenntnisse dieser Studie ist, dass lokale Operatoren die Bilddetails besser beibehalten, wobei globale Operatoren den Kontrast besser darstellen können.

## 4 Algorithmus von Debevec und Malik

Diese Arbeit behandelt im Kern den Ansatz von Debevec und Malik [DM97]. Obwohl der Artikel bereits relativ alt ist (1997), wird das Verfahren in vielen Anwendungen benutzt (siehe Abschnitt 3.1). Dessen Grundgedanke ist es HDR-Bilder aus Bildserien zu generieren, welche mit einer herkömmlichen Kamera-Ausrüstung aufgenommen wurden.

### 4.1 Ansatz

Der Algorithmus schätzt während der Generierung des HDR-Bildes gleichzeitig auch die sog. Antwortkurve der Kamera. Diese Antwortkurve ist die kameraspezifische Abbildung, welche aus den Beleuchtungswerten der aufzunehmenden Szene Grau- bzw. Farbwerte erzeugt (siehe Abbildung 1.1).

Um aus der Belichtungsserie ein HDR-Bild erzeugen zu können, müssen die Beleuchtungswerte der Kamera-Sensorik (hier  $E$ ) identifiziert werden. Normalerweise geschieht dies indem die Umkehrfunktion der Kamera-Antwortkurve vorab berechnet wird. Dazu muss die Kamera durch Test-Bilder vermessen und das System radiometrisch kalibriert werden. Beim Ansatz von Debevec und Malik wird diese Kamera-Antwortfunktion hingegen während der Generierung des HDR-Bildes aus der Belichtungsserie errechnet. Dadurch ist es möglich Belichtungsserien (auch ohne Kenntnisse über die Apparatur) zu HDR-Bildern zu fusionieren.

#### 4.1.1 Verwendete Symbole

In den nachfolgenden Beschreibungen werden analog zu [DM97] folgende Symbole verwendet:

$P$  : Anzahl der Bilder der Bildserie (Anzahl der verschiedenen Belichtungszeiten)

$N$  : Anzahl der Bildpunkte in jedem Bild ( $n \times m$  Bild  $\Rightarrow N = n \cdot m$ )

$Z_{i,j}$  : Grauwert  $i \in [0, N - 1]$  des Bildes  $j \in [0, P - 1]$

$Z_{min}$  : Minimaler Grauwert  $Z_{min} = \min\{Z_{ij}\} \forall i, j$  (wird zur Vereinfachung mit 0 belegt)

$Z_{max}$  : Maximaler Grauwert  $Z_{max} = \max\{Z_{ij}\} \forall i, j$  (wird zur Vereinfachung mit 255 belegt)

$E_i$  : Beleuchtungsstärke im Pixel  $i \in [0, N - 1]$

$F_i$  : Abkürzende Notation für  $\ln E_i$

- $\Delta t_j$  : Belichtungsdauer des Bildes  $j \in [0, P - 1]$
- $f(X)$  : Nichtlineare Funktion, Belichtung  $X$ , Grauwertbild  $Z$ :  $f(X) = Z$
- $\mathbf{g}(z)$  : Kamera-Antwortkurve als diskreter Vektor (abuse of notation)
- $\mathbf{g}'(z)$  : Diskrete Approximation der ersten Ableitung von  $\mathbf{g}$   
 $\mathbf{g}'(z) = \mathbf{g}(z) - \mathbf{g}(z - 1)$
- $\mathbf{g}''(z)$  : Diskrete Approximation der zweiten Ableitung von  $\mathbf{g}$   
 $\mathbf{g}''(z) = \mathbf{g}(z - 1) - 2\mathbf{g}(z) + \mathbf{g}(z + 1)$

### 4.1.2 Herleitung

Da aus physikalischer Sicht angenommen werden kann, dass  $f$  monoton steigend und stetig ist, sei auch  $f^{-1}$  definiert. Damit kann die Belichtung  $X$  mit  $f^{-1}(Z) = X$  berechnet werden. Die Belichtung hängt linear von der Beleuchtungsstärke  $E$  und der Belichtungsdauer  $\Delta t$  mit  $X = E \cdot \Delta t$  ab. Damit ergeben sich die nachfolgenden Zusammenhänge:

$$\begin{aligned}
 Z_{ij} &= f(X_{ij}) \\
 Z_{ij} &= f(E_i \cdot \Delta t_j) && \text{(siehe oben)} \\
 f^{-1}(Z_{ij}) &= E_i \cdot \Delta t_j && \text{(mit Monotonie begründete Umkehrfunktion)} \\
 \ln f^{-1}(Z_{ij}) &= \ln E_i + \ln \Delta t_j && \text{(natürlicher Logarithmus)} \\
 \mathbf{g}(Z_{ij}) &= \ln f^{-1}(Z_{ij}) = \ln E_i + \ln \Delta t_j && \text{(vereinfachte Definition)} \\
 \mathbf{g}(Z_{ij}) &= F_i + \ln \Delta t_j && \text{(abkürzende Notation)}
 \end{aligned}$$

Das aus obiger Gleichung ermittelbare Gleichungssystem hat die Unbekannten  $\mathbf{g}(z)$  und  $\mathbf{F}$ . Um das Gesamtsystem zu lösen und das HDR-Bild zu erzeugen stellen Debevec und Malik folgende Energiefunktion auf, welche minimiert werden muss:

$$\Omega = \underbrace{\sum_{i=1}^N \sum_{j=1}^P [\mathbf{g}(Z_{ij}) - F_i - \ln \Delta t_j]^2}_{\text{Datenterm}} + \lambda \underbrace{\sum_{z=Z_{\min}+1}^{Z_{\max}-1} \mathbf{g}''(z)^2}_{\text{Glattheitsterm}} \quad (4.1)$$

Der Datenterm dieser Energiefunktion sorgt für eine Verknüpfung der Bilder an jedem Pixel und ein Erhalten der Farb- bzw. Grauwerte. Der Glattheitsterm hingegen sorgt für eine möglichst lineare und im Bezug zur zweiten Ableitung glatten Antwortkurve. Das hier (und in den folgenden Gleichungen) verwendete  $z$  im Glattheitsterm ist als diskreter Laufindex zu verstehen.



### 4.1.3 Eindeutigkeit der Lösung für $\mathbf{g}$

Durch die Minimierung der Energiefunktion aus Gleichung 4.1 können  $\mathbf{g}$  und  $\mathbf{F}$  nur auf eine globale Verschiebung  $\alpha$  bestimmt werden. Dies ist daran ersichtlich, dass ein Ersetzen von  $F_i$  durch  $F_i + \alpha$  und  $\mathbf{g}$  durch  $\mathbf{g} + \alpha$  keine Änderung in Gleichung 4.1 hervorrufen würde. Um jedoch eindeutige Ergebnisse für die Antwortkurve  $\mathbf{g}$  und damit auch für  $\mathbf{F}$  zu erhalten wird eine weitere Bedingung für  $\mathbf{g}$  hinzugefügt, die besagt, dass der mittlere Grauwert  $Z_{mid} = \frac{1}{2} \cdot (Z_{min} + Z_{max})$  auch eine bestimmte Belichtung erhalten soll, nämlich  $\mathbf{g}(Z_{mid}) = 0$ . Damit ist die Antwortkurve  $\mathbf{g}$  und der Vektor  $\mathbf{F}$  eindeutig bestimmbar.

## 4.2 Berechnung der Antwortkurve

Aus der Energiefunktion (siehe Gleichung 4.1) lassen sich durch partielles Ableiten nach  $F_i$  und  $\mathbf{g}(k) \quad \forall k \in [Z_{min}, Z_{max}]$  mehrere Gleichungen erstellen. Debevec und Malik schlagen vor, dieses überbestimmte Lineare Gleichungssystem (LGS) mithilfe der Singular Value Decomposition (dt. Singulärwertzerlegung) (SVD) zu lösen. Da das entstehende LGS nur sehr dünn besetzt ist, kann dies mit geringem Rechenaufwand realisiert werden. Für das Aufstellen des Gleichungssystems werden u.a. die Approximation durch zentrale Differenzen für die zweite Ableitung  $\mathbf{g}''(z) = \mathbf{g}(z-1) - 2\mathbf{g}(z) + \mathbf{g}(z+1)$  und die Zusatzbedingung für die Fixierung der Kurve, d.h.  $\mathbf{g}(Z_{mid}) = 0$ , verwendet (siehe Unterabschnitt 4.1.3).

## 4.3 Konstruktion der Radiance Map

Sobald die Antwortkurve  $\mathbf{g}$  bestimmt wurde, kann mit ihrer Hilfe die Radiance Map der Belichtungsserie bestimmt werden. Dies geschieht mittels der Gleichung 4.2, welche wie folgt nach  $F_i$  umgestellt werden kann:

$$g(Z_{ij}) = F_i + \ln \Delta t_j \quad (4.2)$$

$$F_i = \mathbf{g}(Z_{ij}) - \ln \Delta t_j \quad (4.3)$$

Aus Gründen der Robustheit und um alle Bilder bei der Konstruktion der Radiance Map zu verwenden, schlagen Debevec und Malik des Weiteren vor, für die Berechnung von  $F_i$  alle Bilder der Belichtungsserie zu verwenden und diese gewichtet zu mitteln. Dadurch ergibt sich folgende Berechnungsvorschrift für  $F_i$ :

$$F_i = \frac{\sum_{j=1}^P w^2(Z_{ij}) \cdot (\mathbf{g}(Z_{ij}) - \ln \Delta t_j)}{\sum_{j=1}^P w^2(Z_{ij})} \quad (4.4)$$

Nachdem nun das Verfahren zur Berechnung der HDR-Bilder bekannt ist, werden im nächsten Abschnitt die möglichen Erweiterungen diskutiert.

## 4.4 Mögliche Erweiterungen des Ansatzes

Der grundlegende Ansatz von Debevec und Malik (Gleichung 4.1) hat einige Schwachstellen. Diese werden zum Teil bereits von den Autoren des Artikels angesprochen und werden hier der Vollständigkeit halber aufgelistet.

### 4.4.1 Gewichtungsfunktion

Den Belichtungswert für die Bereiche nahe  $Z_{min}$  und  $Z_{max}$  zu berechnen ist eine Herausforderung, da diese Pixel häufig saturiert sind und damit der tatsächliche Wert nicht bekannt ist. Hinzu kommt, dass  $\mathbf{g}$  typischerweise sehr steil in der Nähe von  $Z_{min}$  und  $Z_{max}$  sein wird. Aus diesen beiden Gründen ist es sinnvoll, diese Randbereiche bei der Berechnung von  $\mathbf{g}$  weniger stark zu gewichten. Aus diesem Grund wird eine Gewichtungsfunktion  $w(z)$  als Dreiecksfunktion eingeführt, die wie folgt definiert wird:

$$w(z) = \begin{cases} z - Z_{min} & \text{falls } z \leq Z_{mid} \\ Z_{max} - z & \text{sonst} \end{cases} \quad (4.5)$$

Durch diese werden die Terme der Energiefunktion in der Mitte stärker und die steilen äußeren Bereiche der Kurve  $\mathbf{g}$  weniger gewichtet. Diese Gewichtungsfunktion wird außerdem auch bei der Rekonstruktion der Radiance Map verwendet, um den Einfluss der Bildpunkte über die gesamte Belichtungsserie zu mitteln. Dies führt zu den nachfolgenden Veränderungen der ursprünglichen Energiefunktion (siehe Gleichung 4.6):

$$\Omega = \sum_{i=1}^N \sum_{j=1}^P w^2(Z_{ij}) \cdot [\mathbf{g}(Z_{ij}) - F_i - \ln \Delta t_j]^2 + \lambda \sum_{z=Z_{min}+1}^{Z_{max}-1} [w(z) \cdot \mathbf{g}''(z)]^2 \quad (4.6)$$

### 4.4.2 Selektion von Bildpunkten

Debevec und Malik stellen fest, dass bei der Schätzung der Kamera-Antwortkurve nicht jeder Pixel in den Ausgangsbildern verwendet werden muss. Das von ihnen vorgestellte Verfahren führt zu einem LGS mit  $N \times P + Z_{min} - Z_{max}$  Unbekannten. Um das Gleichungssystem ausreichend überbestimmt zu halten, schlagen sie deswegen vor,  $N$  so zu wählen, dass  $N \cdot (P - 1) > (Z_{max} - Z_{min})$  gilt. Hierbei soll  $N$  nicht wesentlich größer als  $\frac{(Z_{max} - Z_{min})}{P - 1}$  sein, damit die Anzahl der Unbekannten möglichst gering bleibt.

Nur durch die Reduktion der betrachteten Pixel kann das LGS effizient gelöst werden. Jedoch wird dadurch auch die verwendete Information aus den Bildern reduziert und somit kann es zu Abweichungen der geschätzten von der tatsächlichen Antwortkurve kommen. Während des eigentlichen Verfahrens zur Berechnung der Antwortkurve wird  $F_i$  nur für die selektierten Pixel berechnet. Für die übrigen Bildpunkte geschieht dies erst nach Abschluss des Verfahrens unter Verwendung der berechneten Antwortkurve  $\mathbf{g}$ .

Diese Selektion der Referenzpunkte aus den Belichtungsserien wird von Debevec und Malik noch händisch durchgeführt. Bei elf Bildern in einer Belichtungsreihe schlagen sie vor ca. 50 Bildkoordinaten zu bestimmen, die für die Berechnung verwendet werden sollen. Dabei ist darauf zu achten, dass diese Koordinaten gleichmäßig über die Ausgangsbilder verteilt sind und dass sie aus Regionen stammen, die keine große Varianz aufweisen. Dies macht die Schätzung der Antwortkurve anfällig für Rauschen auf dem Ausgangsmaterial und soll damit verhindert werden. Einen Ansatz zum automatisierten Festlegen der Bildpunkte stellen sie nicht vor.

Darüber hinaus kann das Verfahren somit auch keine Forderungen an die resultierende Radiance Map stellen, da nicht alle Bildpunkte in die Berechnung der Kamera-Antwortkurve mit einbezogen werden.

### 4.4.3 Robustheit des Verfahrens

In vielen Bildbearbeitungs-Algorithmen werden heutzutage robuste Funktionen eingesetzt, um Messfehler und Rauschen weniger stark zu gewichten. Die übliche quadratische Bestrafung in Datentermen mit  $\varphi(s^2) = s^2$  ist in Bezug auf große Fehler und Modellabweichungen nicht robust. Eine typische Erweiterung ergibt sich durch den Einsatz subquadratischer Bestrafungsfunktionen [Bruo6, S. 9f, S. 87f]. Diese haben den Vorteil, dass Ausreißer in der Eingabe (wie z.B. Messfehler oder Rauschen) bei der Minimierung abgeschwächt werden und somit das Ergebnis weniger stark beeinflusst wird. In unserem Fall verwenden wir folgende differenzierbare Approximation der Betragsfunktion:

$$\varphi(s^2) = \sqrt{s^2 + \epsilon^2} \tag{4.7}$$

Bei  $\epsilon$  handelt es sich um einen betragsmäßig kleinen Wert, der dafür sorgt, dass die Funktion nicht linear ist. So bleiben notwendige Eigenschaften bezüglich der Differenzierbarkeit erhalten.

### 4.4.4 Monotonie-Kriterium

Aus physikalischer und radiometrischer Sicht muss die Kamera-Antwortkurve (streng) monoton steigend sein. Diese Eigenschaft wird für  $\mathbf{g}$  im Standard-Ansatz nicht weiter verfolgt und wird durch den dort verwendeten Glattheitsterm auch nicht notwendiger Weise

sicher gestellt. Ein Bestandteil dieser Arbeit ist deshalb auch, im Verfahren eine Forderung an die Monotonie von  $\mathbf{g}$  zu ergänzen und dieses erweiterte Verfahren zu implementieren (siehe Abschnitt 5.2).

## 5 Mathematische Ausarbeitung

Zur Lösung des LGS aus Gleichung 4.6 soll in der vorliegenden Arbeit ein anderes Verfahren verwendet werden. Hierbei wird das Lösen nach  $\mathbf{g}$  und  $F_i$  in zwei Teilprobleme zerlegt, welche alternierend gelöst werden. Der Vorteil dieses Ansatzes ist, dass die gesamten Bildinformationen aus der Belichtungsserie verwendet werden können. Dies ist möglich, da die entstehenden Gleichungssysteme sehr dünn besetzt sind und effizient gelöst werden können. Die Struktur des alternierenden Vorgehens ist in Algorithmus 5.1 beschrieben.

Dieser Algorithmus dient dazu, aus einer bestehenden Zwischenlösung immer eine neue Näherung der anderen Unbekannten zu berechnen, solange bis das Resultat konvergiert. Bei gegebenem  $\mathbf{g}$  wird eine neue Instanz von  $\mathbf{F}$  berechnet, die dann wieder für die Schätzung von  $\mathbf{g}$  verwendet wird.

Darüber hinaus kann durch dieses Verfahren neben der Schätzung der Kamera-Antwortkurve auch gleichzeitig die Radiance Map des HDR-Bildes mit berechnet werden. Dadurch spart man sich die anschließende Umrechnung der Bildpunkte mittels der Funktion  $\mathbf{g}$  und ist außerdem in der Lage Forderungen an  $\mathbf{F}$  zu stellen.

In den nachfolgenden Abschnitten werden häufig Approximationen für die erste und zweite Ableitung verwendet. Dass es sich hierbei deswegen in der Regel um keine exakte Gleichheit ( $=$ ) handelt, sondern vielmehr um eine Annäherung ( $\approx$ ) sei hier erwähnt. Es wird im Nachfolgenden aus Gründen der Lesbarkeit darauf verzichtet, dies kenntlich zu machen.

### 5.1 Optimierungsansatz

Die Gleichung 4.6 dient als Grundlage für den Optimierungsansatz des gesamten Verfahrens. Da diese Energiefunktion minimiert werden soll, sind partielle Ableitungen nach  $\mathbf{g}(k)$  bzw.  $F_i$

---

**Algorithmus 5.1** Alternierendes Lösen nach  $g(k)$  und  $F_i$

---

```
function SOLVEHDR( $Z_{ij}, \ln \Delta t_j, N, P$ )  
   $g \leftarrow \text{init}G()$   
  while  $g$  changes do  
     $\mathbf{F} \leftarrow \text{solve}F(\mathbf{F}, \mathbf{g}, Z_{ij}, \ln \Delta t_j, N, P)$   
     $\mathbf{g} \leftarrow \text{solve}G(\mathbf{F}, \mathbf{g}, Z_{ij}, \ln \Delta t_j, N, P)$   
  end while  
  return [ $\mathbf{g}, \mathbf{F}$ ]  
end function
```

---

notwendig. Die vorkommenden Ableitungen zweiter Ordnung werden mittels der zentralen Differenz  $\mathbf{g}''(k) = \mathbf{g}(k-1) - 2\mathbf{g}(k) + \mathbf{g}(k+1)$  diskretisiert. Dies führt zu nachfolgender Umstellung der Energiefunktion:

$$\Omega = \sum_{i=0}^{N-1} \sum_{j=0}^{P-1} w^2(Z_{ij}) \cdot [\mathbf{g}(Z_{ij}) - F_i - \ln \Delta t_j]^2 + \lambda \sum_{z=Z_{min}+1}^{Z_{max}-1} [w(z) \cdot \mathbf{g}''(z)]^2 \quad (5.1)$$

$$= \underbrace{\sum_{i=0}^{N-1} \sum_{j=0}^{P-1} w^2(Z_{ij}) \cdot [\mathbf{g}(Z_{ij}) - F_i - \ln \Delta t_j]^2}_{\Phi} + \lambda \underbrace{\sum_{z=Z_{min}+1}^{Z_{max}-1} w^2(z) \cdot [\mathbf{g}(z-1) - 2\mathbf{g}(z) + \mathbf{g}(z+1)]^2}_{\Theta} \quad (5.2)$$

Diskretisierung von  $\mathbf{g}''(k)$

In den folgenden Herleitungen taucht häufig der Faktor 2 auf. Dieser entsteht durch das Ableiten der quadratischen Bestrafungsfunktionen. Er taucht in der Regel in allen Summanden von  $\partial\Omega$  auf und kann deswegen gekürzt werden. In besonderen Fällen (wie z.B. der Erweiterung um robuste Bestrafungsterme, siehe Abschnitt 5.4) ist dies nicht der Fall. Dann werden diese Faktoren separat behandelt.

### 5.1.1 Gleichungssystem für $\mathbf{g}$

Um nun das LGS zur Lösung nach  $\mathbf{g}$  aufzustellen, muss  $\Omega$  zunächst partiell nach  $\mathbf{g}(k) \forall k \in [0, 255]$  abgeleitet werden. Hierbei kann man die Ableitungen nach  $\Phi$  und  $\Theta$  zunächst separat betrachten:

$$\frac{\partial\Omega}{\partial\mathbf{g}(k)} = \frac{\partial\Phi}{\partial\mathbf{g}(k)} + \frac{\partial\Theta}{\partial\mathbf{g}(k)} \quad (5.3)$$

Bei der Ableitung von  $\Phi$  kommt eine Einschaltfunktion  $\delta_{z=k}$  zum Einsatz, die wie folgt definiert ist:

$$\delta_{z=k} = \begin{cases} 1 & \text{wenn } z = k \\ 0 & \text{sonst} \end{cases} \quad (5.4)$$

Unter Zuhilfenahme dieser Funktion kann die partielle Ableitung von  $\Phi$  wie folgt notiert werden:

$$\frac{\partial \Phi}{\partial \mathbf{g}(k)} = 2 \cdot w^2(k) \cdot \sum_{i=0}^{N-1} \sum_{j=0}^{P-1} [\mathbf{g}(k) - F_i - \ln \Delta t_j] \cdot \delta_{Z_{ij}=k} \quad (5.5)$$

$$= 2 \cdot w^2(k) \cdot \mathbf{g}(k) \sum_{i=0}^{N-1} \sum_{j=0}^{P-1} \delta_{Z_{ij}=k} - 2 \cdot w^2(k) \cdot \sum_{i=0}^{N-1} \sum_{j=0}^{P-1} \delta_{Z_{ij}=k} (F_i - \ln \Delta t_j) \quad (5.6)$$

$$= \underbrace{w^2(k) \sum_{i=0}^{N-1} \sum_{j=0}^{P-1} (\delta_{Z_{ij}=k}) \cdot \mathbf{g}(k)}_{\text{Matrixeintrag } a_k} - \underbrace{w^2(k) \sum_{i=0}^{N-1} \sum_{j=0}^{P-1} (F_i - \ln \Delta t_j) \delta_{Z_{ij}=k}}_{\text{Vektoreintrag } b_k} \quad (5.7)$$

Dieses System kann in Matrixschreibweise umformuliert werden, wobei die Koeffizienten der Matrix aus den einzelnen partiellen Ableitungen gewonnen werden können:

$$\begin{pmatrix} \ddots & 0 & 0 \\ 0 & a_k & 0 \\ 0 & 0 & \ddots \end{pmatrix} \cdot \begin{pmatrix} \vdots \\ \mathbf{g}(k) \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ b_k \\ \vdots \end{pmatrix} \quad (5.8)$$

Anschließend betrachten wir den Glattheitsterm  $\Theta$ . Auch dieser muss partiell nach  $\mathbf{g}(k)$  abgeleitet werden. Aus Gründen der Vereinfachung wurden in den folgenden Berechnungen  $Z_{min} = 0$  und  $Z_{max} = 255$  angenommen. In der Gleichung 5.2 wurde der Gewichtungsfaktor für den Glattheitsterm  $\lambda$  absichtlich nicht in  $\Theta$  eingebunden, da dieser bei der Herleitung keine Rolle spielt. Der Faktor  $\lambda$  wird am Ende wieder hinzugefügt. Bei der partiellen Ableitung des Glattheitsterms muss hier besonders auf die Randbedingungen geachtet werden, dort verhalten sich die partiellen Ableitungen anders:

$$\frac{\partial \Theta}{\partial \mathbf{g}(0)} = w^2(1) \cdot \mathbf{g}(0) - 2w^2(1) \cdot \mathbf{g}(1) + w^2(1) \cdot \mathbf{g}(2) \quad (5.9)$$

$$\begin{aligned} \frac{\partial \Theta}{\partial \mathbf{g}(1)} &= -2w^2(1) \cdot \mathbf{g}(0) \\ &+ [4w^2(1) + w^2(2)] \cdot \mathbf{g}(1) \\ &- 2[w^2(1) + w^2(2)] \cdot \mathbf{g}(2) \\ &+ w^2(2) \cdot \mathbf{g}(3) \end{aligned} \quad (5.10)$$

$$\begin{aligned} \frac{\partial \Theta}{\partial \mathbf{g}(k)} &= w^2(k-1) \cdot \mathbf{g}(k-2) \\ &- 2[w^2(k-1) + 2w^2(k)] \cdot \mathbf{g}(k-1) \\ &+ [w^2(k-1) + 4w^2(k) + w^2(k+1)] \cdot \mathbf{g}(k) \\ &- 2[w^2(k) + w^2(k+1)] \cdot \mathbf{g}(k+1) \\ &+ w^2(k+1) \cdot \mathbf{g}(k+1) \quad \forall k \in [2, 253] \end{aligned} \quad (5.11)$$

$$\begin{aligned} \frac{\partial \Theta}{\partial \mathbf{g}(254)} &= w^2(253) \cdot \mathbf{g}(252) \\ &\quad - 2(w^2(253) + w^2(254)) \cdot \mathbf{g}(253) \\ &\quad + (w^2(253) + 4w^2(254)) \cdot \mathbf{g}(254) \\ &\quad - 2w^2(254) \cdot \mathbf{g}(255) \end{aligned} \quad (5.12)$$

$$\frac{\partial \Theta}{\partial \mathbf{g}(255)} = w^2(254) \cdot \mathbf{g}(253) - 2w^2(254) \cdot \mathbf{g}(254) + w^2(254) \cdot \mathbf{g}(255) \quad (5.13)$$

Auch diese Gleichungen lassen sich wieder in Matrixschreibweise umformulieren (siehe Gleichung 5.14). Die Koeffizienten der Matrix gehen aus obigen Gleichungen hervor (z.B.  $d_{0,0} = w^2(1)$ ,  $d_{1,-1} = -2w^2(1), \dots$ ). Der Faktor  $\lambda$  wurde hier wieder mit hineingezogen (siehe Gleichung 5.2). Bei der Matrix  $D_4$  handelt es sich um eine diskrete Approximation der vierten Ableitung.

$$\lambda \underbrace{\begin{pmatrix} d_{0,0} & d_{0,1} & d_{0,2} & 0 & \dots & & & & \\ d_{1,-1} & d_{1,0} & d_{1,1} & d_{1,2} & 0 & \dots & & & \\ & & \ddots & & & & & & \\ \dots & d_{k,-2} & d_{k,-1} & d_{k,0} & d_{k,1} & d_{k,2} & \dots & & \\ & & & & \ddots & & & & \\ & & & d_{254,-2} & d_{254,-1} & d_{254,0} & d_{254,1} & & \\ & & & & d_{255,-2} & d_{255,-1} & d_{255,0} & & \end{pmatrix}}_{\text{Matrix } D_4} \cdot \begin{pmatrix} \vdots \\ \vdots \\ \vdots \\ \mathbf{g}(k) \\ \vdots \\ \vdots \end{pmatrix} \quad (5.14)$$

Die Gleichungen 5.7 und 5.9 – 5.13 (bzw. die Matrixschreibweisen aus den Gleichungen 5.8 und 5.14) ergeben zusammen mit der Ausgangssituation (siehe Gleichung 5.3) das folgende endgültige Gleichungssystem für  $\mathbf{g}$ , welches ebenfalls in Matrix-Notation aufgestellt werden kann:

$$\underbrace{[A + \lambda D_4]}_{\text{Matrix M}} \cdot \mathbf{g} = b \quad (5.15)$$

Zu beachten ist, dass  $M$  eine Pentadiagonalmatrix ist. Dies kann beim Lösen des LGS genutzt werden, indem eine spezialisierte Variante der LU-Zerlegung verwendet wird (siehe Unterabschnitt 5.5.1).

Außerdem führt die Zerlegung des Problems in das separierte Lösen nach  $\mathbf{g}$  und  $\mathbf{E}$  dazu, dass die ursprünglich erwähnte Eigenschaft der unendlichen Anzahl von Lösungen (siehe Unterabschnitt 4.1.3) nicht mehr besteht, da die beiden Schritte des Verfahrens separat und alternierend ausgeführt werden und immer eine konkrete Näherungslösung der jeweils



anderen Unbekannten vorliegt, die implizit eine (unbekannte) Verschiebung enthält. Um diese Problematik zu umgehen, wird deshalb nach der Berechnung von  $\mathbf{g}$  die Kurve immer so verschoben, dass  $\mathbf{g}(Z_{mid}) = 0$  gilt. Diese Verschiebung wird durch die folgende Vorschrift durchgeführt:

$$\tilde{\mathbf{g}}(k) = \mathbf{g}(k) - \mathbf{g}(Z_{mid}) \quad \forall k \in [0, 255] \quad (5.16)$$

Damit ist sichergestellt, dass alle Antwortkurven, die durch das Verfahren berechnet werden, vergleichbar und eindeutig sind.

### 5.1.2 Lösen nach F

Im Gegensatz zu  $\mathbf{g}$  kann  $\mathbf{F}$  ohne ein LGS berechnet werden, da es punktweise definiert ist. Die Berechnungsvorschrift für  $F_i$  resultiert aus der partiellen Ableitung der Energiefunktion aus Gleichung 5.2 nach  $F_i$  und lautet:

$$F_i = \frac{\sum_{j=0}^{P-1} (\mathbf{g}(Z_{ij}) - \ln \Delta t_j) \cdot w^2(Z_{ij})}{\sum_{j=0}^{P-1} w^2(Z_{ij})} \quad (5.17)$$

Mithilfe dieser Funktion kann nun nach errechnetem  $\mathbf{g}$  auch die neue Instanz von  $\mathbf{F}$  berechnet werden, die dann im Zuge des alternierenden Verfahrens wieder Eingabe für die Berechnung von  $\mathbf{g}$  ist.

Nach der grundsätzlichen Vorstellung des Verfahrens sollen nun die möglichen Erweiterungen beschrieben und formal spezifiziert werden. Dazu wird das Standard-Verfahren schrittweise erweitert.

## 5.2 Erweiterung um Monotonie-Eigenschaft

Im Ansatz von Debevec und Malik wird für die Funktion  $f$  angenommen, dass sie monoton und damit invertierbar ist. Für die diskrete Funktion  $\mathbf{g}$  wird dies jedoch nicht explizit gefordert. Aus physikalischer und radiometrischer Sicht muss  $\mathbf{g}$  jedoch ebenfalls (streng) monoton sein. Deshalb wird nachfolgend eine erste Erweiterung des Algorithmus mit einer Forderung an die Monotonie von  $\mathbf{g}$  vorgestellt. Dies lässt sich über die Energiefunktion  $\Omega$  als weiteren Bestrafungsterm  $\Gamma$  realisieren. Damit erhalten wir die neue Funktion  $\tilde{\Omega}$ :

$$\tilde{\Omega} = \Omega + \underbrace{\mu \sum_{z=1}^{255} w^2(z) [(\phi_{\mathbf{g}' < 0}(z) \cdot \mathbf{g}'(z))]^2}_{\text{Monotonie-Forderung } \Gamma} = \Omega + \Gamma \quad (5.18)$$

Die in dieser Gleichung vorkommende Funktion  $\phi_{\mathbf{g}' < 0}(z)$  ist eine Indikatorfunktion, die immer dann aktiv wird, wenn die Monotonie der Kurve  $\mathbf{g}$  an einer Stelle verletzt ist, und wird folgendermaßen definiert:

$$\phi_{\mathbf{g}' < 0}(z) = \begin{cases} 1, & \text{falls } \mathbf{g}'(z) < 0 \\ 0 & \text{sonst} \end{cases} \quad (5.19)$$

Hier wird ebenfalls der Least-Squares-Ansatz (quadratische Bestrafungsfunktion) verwendet, welcher auch schon im Standard-Verfahren zum Einsatz kommt (siehe Gleichung 4.6). Dadurch werden Abweichungen von der Monotonie quadratisch bestraft.

Da  $\mathbf{g}'(z)$  bei der Berechnung von  $\mathbf{g}$  nicht bekannt ist, wird für die Indikatorfunktion die Instanz  $\mathbf{g}$  aus der vorherigen Iteration verwendet. Durch die Diskretisierung mit der einseitigen finiten Differenz  $\mathbf{g}'(z) = \mathbf{g}(z) - \mathbf{g}(z - 1)$  erhält man dadurch den zusätzlichen Summanden  $\Gamma$ :

$$\Gamma = \mu \sum_{z=1}^{255} w^2(z) \cdot \phi_{\mathbf{g}' < 0}^2(z) \cdot (\mathbf{g}(z) - \mathbf{g}(z - 1))^2 \quad (5.20)$$

Auch dieser muss im Zuge der Minimierung der Energiefunktion partiell nach  $\mathbf{g}(k)$  abgeleitet werden, was zu folgenden Gleichungen führt:

$$\frac{\partial \Gamma}{\partial \mathbf{g}(0)} = -2\mu w^2(1) \cdot \phi_{\mathbf{g}' < 0}^2(1) \cdot (\mathbf{g}(1) - \mathbf{g}(0)) \quad (5.21)$$

$$\begin{aligned} \frac{\partial \Gamma}{\partial \mathbf{g}(k)} = & 2\mu w^2(k) \cdot \phi_{\mathbf{g}' < 0}^2(k) \cdot (\mathbf{g}(k) - \mathbf{g}(k - 1)) \\ & - 2\mu w^2(k + 1) \cdot \phi_{\mathbf{g}' < 0}^2(k + 1) \cdot (\mathbf{g}(k + 1) - \mathbf{g}(k)), \quad \forall k \in [1, 254] \end{aligned} \quad (5.22)$$

$$\frac{\partial \Gamma}{\partial \mathbf{g}(255)} = -2\mu w^2(255) \cdot \phi_{\mathbf{g}' < 0}^2(255) \cdot (\mathbf{g}(255) - \mathbf{g}(254)) \quad (5.23)$$

Aus den Gleichungen 5.21 – 5.23 lässt sich nun wieder eine Matrix mit folgender Stuktur erzeugen (hier wurde  $\phi_{\mathbf{g}'<0}^2(k) = \phi_{\mathbf{g}'}^2(k)$  zur Kürzung verwendet):

$$2\mu \underbrace{\begin{pmatrix} -w^2(1)\phi_{\mathbf{g}'}^2(1) & w^2(1)\phi_{\mathbf{g}'}^2(1) & 0 & \dots & & \\ & \ddots & & & & \\ \dots & -w^2(k)\phi_{\mathbf{g}'}^2(k) & w^2(k)\phi_{\mathbf{g}'}^2(k) & -w^2(k+1)\phi_{\mathbf{g}'}^2(k+1) & \dots & \\ & & +w^2(k+1)\phi_{\mathbf{g}'}^2(k+1) & & & \\ & & \ddots & & & \\ \dots & & 0 & w^2(255)\phi_{\mathbf{g}'}^2(255) & -w^2(255)\phi_{\mathbf{g}'}^2(255) & \end{pmatrix}}_{\text{Matrix } c} \quad (5.24)$$

Diese Berechnung kann auch über Matrizenmultiplikation erreicht werden.  $D$  ist dabei eine Matrix, welche die erste Ableitung approximiert.  $V$  ist eine Diagonal-Matrix mit  $v_{i,i} = \phi_{\mathbf{g}'<0}(i)$ .  $W$  ist die Diagonal-Matrix der Gewichte mit  $w_{i,i} = w(i)$ . Die damit entstehende Gleichung 5.25 kann dann partiell zur Gleichung 5.26 abgeleitet werden.

$$\Gamma = \mu \cdot (WVD\mathbf{g})^2 = \mu \cdot \mathbf{g}^T \cdot D^T V^T W^T WVD \cdot \mathbf{g} \quad (5.25)$$

$$\frac{\partial \Gamma}{\partial \mathbf{g}} = 2 \cdot \mu \cdot \underbrace{D^T V^T W^T WVD}_{\text{Matrix } c} \cdot \mathbf{g} \quad (5.26)$$

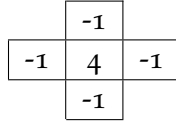
Der Parameter  $\mu$  ist ähnlich wie  $\lambda$  ein Gewichtungsfaktor für die Monotonie-Bedingung. Das Gleichungssystem aus 5.15 kann damit um den Summanden aus Gleichung 5.26 ergänzt werden.

$$[M + \mu C] \cdot \mathbf{g} = b \quad (5.27)$$

Zu beachten ist, dass die Matrix  $C$  ebenfalls pentadiagonal ist und somit auch bei diesem LGS die besondere Eigenschaft bestehen bleibt. Deshalb kann auch hier die LU-Zerlegung einer Pentadiagonalmatrix (siehe Unterabschnitt 5.5.1) verwendet werden.

### 5.3 Erweiterung um einen räumlichen Glattheitsterm

Die Erweiterung um den räumlichen Glattheitsterm (also eine Forderung an  $F$  sich im zweidimensionalen Bild möglichst glatt zu verhalten) ist eine sinnvolle Erweiterung, um die Berechnung der  $F_i$  in Bezug auf Rauschen zu verbessern. Dieses Rauschen kann z.B. bei niedrigen Belichtungszeiten und einer damit geringen Belichtungsintensität (wenig eingetroffene Photonen) entstehen. Ziel dieser Erweiterung ist es daher, das lokale Umfeld



**Abbildung 5.1:** Einfluss der umliegenden Bildpunkte — Räumlicher Glattheitsterm als 2D Nachbarschafts-Maske.

um einen Bildpunkt mit in die Berechnung einzubeziehen. Dazu wird eine räumliche Glattheit gefordert, welche (analog zum Glattheitsterm von  $\mathbf{g}$ ) mittels der ersten Ableitung ausgedrückt werden kann. Da es sich bei  $\mathbf{E}$  um den zweidimensionalen Bildbereich handelt, müssen die Ableitungen in  $x$ - und  $y$ -Richtung betrachtet werden. Der Vektor  $\mathbf{F}$  ist eine eindimensionale Darstellung des zweidimensionalen Bildbereiches ( $n \times m$ ), wobei gilt:  $i = x + y * n$  ( $x \in [0, n - 1]$ ,  $y \in [0, m - 1]$ ). Die Glattheitsforderung wird dabei durch die Approximation der ersten Ableitung durch eine einseitige finite Differenz realisiert, wobei die Ränder speziell behandelt werden:

$$\tilde{\Omega} = \Phi + \Theta + \underbrace{\alpha \sum_{i \in A} \left( \overbrace{F_i - F_{i-1}}^{\text{Abltg. nach } x} \right)^2 + \alpha \sum_{i=n}^{N-1} \left( \overbrace{F_i - F_{i-n}}^{\text{Abltg. nach } y} \right)^2}_{\text{Glattheitsterm } \Psi} \quad (5.28)$$

$$A = \{i \in [0, N - 1]\} \setminus \{i \cdot k | k \in \mathbb{N}_+^0\} \quad (5.29)$$

Auch hier muss nun wieder nach  $F_i$  partiell abgeleitet werden, um das Minimum der Energiefunktion (siehe Gleichung 5.28) zu bestimmen, wobei die Randbedingungen zunächst vernachlässigt werden:

$$\frac{\partial \Psi}{\partial F_i} = 2\alpha[(F_i - F_{i-1}) - (F_{i+1} - F_i) + (F_i - F_{i-n}) - (F_{i+n} - F_i)] \quad (5.30)$$

$$= 2\alpha[4F_i - F_{i-n} - F_{i-1} - F_{i+1} - F_{i+n}] \quad (5.31)$$

Unter vernachlässigten Randbedingungen entsteht also eine Nachbarschafts-Maske, die den Einfluss der umliegenden Bildpunkte angibt (siehe Abbildung 5.1).

Um nun das gesamte Gleichungssystem für  $F_i$  aufzustellen, müssen zunächst noch die Terme  $\Theta$  und  $\Phi$  und ihre partiellen Ableitungen betrachtet werden:

$$\frac{\partial \Theta}{\partial F} = 0 \quad (5.32)$$

$$\Phi = \sum_{i=0}^{N-1} \sum_{j=0}^{P-1} w^2(Z_{ij}) [\mathbf{g}(Z_{ij}) - F_i - \ln \Delta t_j]^2 \quad (5.33)$$

$$\frac{\partial \Phi}{\partial F_k} = 2 \sum_{j=0}^{P-1} w^2(Z_{kj}) [\mathbf{g}(Z_{kj}) - F_k - \ln \Delta t_j] \quad (5.34)$$

$$= 2 \underbrace{\sum_{j=0}^{P-1} w^2(Z_{kj}) [\mathbf{g}(Z_{kj}) - \ln \Delta t_j]}_{\text{Vektoreintrag } b_k} - 2 \underbrace{\sum_{j=0}^{P-1} w^2(Z_{kj})}_{\text{Matrixeintrag } H_{k,k}} F_k \quad (5.35)$$

$$\partial \Phi = 2\mathbf{b} - 2H \cdot \mathbf{F} \quad (5.36)$$

Durch Betrachtung der Randbedingungen kann nun aus den Gleichungen 5.30 und 5.35 die Matrix  $R$  für die Einbindung der räumlichen Glattheitsforderung erstellt werden. Diese ist eine  $N \times N$  Matrix, die in Blöcken der Größe  $n \times m$  aufgeteilt werden kann. Ein solcher Block auf der Diagonalen der Matrix steht jeweils für eine Reihe von Bildpunkten im Bild (siehe Abbildung 5.2).

Auch die Addition der Gleichungen 5.30 und 5.35 kann wieder in Matrixschreibweise notierter werden und ergibt unter Berücksichtigung der Randbedingungen dann das nachfolgende Gleichungssystem für die Berechnung von  $\mathbf{F}$ :

$$2H \cdot \mathbf{F} + 2\alpha R \cdot \mathbf{F} = 2\mathbf{b} \quad (5.37)$$

$$(H + \alpha R) \cdot \mathbf{F} = \mathbf{b} \quad (5.38)$$

Das Gleichungssystem aus 5.38 ist symmetrisch, quadratisch und positiv semidefinit. Aufgrund dieser Eigenschaften kann beim Lösen des LGS das schnell konvergierende SOR-Verfahren (siehe Unterabschnitt 5.5.2) verwendet werden.

## 5.4 Erweiterung um Robustheit

Wie im Unterabschnitt 4.4.3 bereits beschrieben, setzt das Verfahren von Debevec und Malik nur quadratische Bestrafungsterme ein. Diese reduzieren die Auswirkungen von Gauß-Rauschen auf den Eingabebildern. Bei anderen Messungenauigkeiten (wie z.B. Salt & Pepper Rauschen) ist ein subquadratischer Bestrafungsterm aus Sicht der Robustheit des Verfahrens besser geeignet, da hier die starken Ausreißer weniger stark bestrafend wirken.



### 5.4.2 Subquadratische Bestrafungsfunktion im Datenterm von $\mathbf{g}$ und $\mathbf{F}$

Der Datenterm von  $\mathbf{g}$  berücksichtigt bisher keine Ausreißer. Sind also starke Ausreißer in den Bildern der Belichtungsserie zu finden (wie z.B. Rauschen), dann werden diese den Datenterm quadratisch beeinflussen. Besser wäre es hier, große Ausreißer weniger stark zu gewichten. Hier kommen die subquadratischen Bestrafungsfunktionen zum Einsatz, die die Energiefunktion aus Gleichung 4.6 erweitert:

$$\tilde{\Omega} = \underbrace{\sum_{i=0}^{N-1} \sum_{j=0}^{P-1} w^2(Z_{ij}) \cdot \varphi([\mathbf{g}(Z_{ij}) - F_i - \ln \Delta t_j]^2)}_{\text{Datenterm mit Robustheit } \Phi} + \lambda \underbrace{\sum_{z=Z_{min}+1}^{Z_{max}-1} [w(Z_{ij}) \cdot \mathbf{g}''(z)]^2}_{\text{Glattheitsterm für } g} \quad (5.40)$$

Diese wird dann wieder partiell nach  $\mathbf{g}(k)$  und  $F_i$  abgeleitet, um den Optimierungsansatz zu lösen:

$$\frac{\partial \tilde{\Phi}}{\partial \mathbf{g}(k)} = 2w^2(k) \sum_{i=0}^{N-1} \sum_{j=0}^{P-1} \delta_{Z_{ij}=k} \underbrace{\varphi'([\mathbf{g}(k) - F_i - \ln \Delta t_j]^2)}_{\text{wird festgehalten}} (\mathbf{g}(k) - F_i - \ln \Delta t_j) \quad (5.41)$$

$$\begin{aligned} &= 2w^2(k) \underbrace{\sum_{i=0}^{N-1} \sum_{j=0}^{P-1} \delta_{Z_{ij}=k} \varphi'([\mathbf{g}(k) - F_i - \ln \Delta t_j]^2)}_{\text{Matrixeintrag } \tilde{a}_k} \mathbf{g}(k) \\ &\quad - 2w^2(k) \underbrace{\sum_{i=0}^{N-1} \sum_{j=0}^{P-1} (F_i + \ln \Delta t_j) \varphi'([\mathbf{g}(k) - F_i - \ln \Delta t_j]^2) \delta_{Z_{ij}=k}}_{\text{Vektoreintrag } \tilde{b}_k} \end{aligned} \quad (5.42)$$

Das so entstehende Gleichungssystem ist durch die von  $\mathbf{g}$  und  $\mathbf{F}$  abhängigen Faktoren  $\varphi'$  nicht mehr linear. Um das nichtlineare Gleichungssystem in eine Serie von linearen Gleichungssystemen zu übertragen, werden die vorkommenden Bestrafungsfaktoren  $\varphi'([\mathbf{g}(k) - F_i - \ln \Delta t_j]^2)$  festgehalten und nach jedem Schritt neu aus den alten Werten von  $\mathbf{g}$  und  $F_i$  berechnet.

Durch diesen Schritt kann das nichtlineare Gleichungssystem auch weiterhin mit der bereits beschriebenen LU-Zerlegung (siehe Unterabschnitt 5.5.1) gelöst werden, jedoch werden nun innere Iterationen des Verfahrens benötigt (siehe Algorithmus 5.2).

Die Koeffizienten  $\tilde{a}_k$  und  $\tilde{b}_k$  ersetzen dabei die Matrix- bzw. Vektoreinträge des Gleichungssystems aus Gleichung 5.8. Der Glattheitsterm  $\Theta$  bleibt zusammen mit seinen partiellen Ableitungen identisch.

Diese Erweiterung muss nun auch noch bei der Berechnung von  $F_i$  berücksichtigt werden. Auch hierzu wird die Energiefunktion  $\tilde{\Omega}$  (siehe Gleichung 5.40) wieder partiell nach  $F_i$  abgeleitet. Aus der Gleichung 5.17 entsteht dann bei aktiviertem subquadratischen Bestrafungsterm die neue Berechnung von  $F_i$ :

**Algorithmus 5.2** Erweitertes alternierendes Lösen nach  $g(k)$  und  $F_i$  mit Haupt- und Inneniterationen. Das nichtlineare Gleichungssystem wurde in eine Serie von linearen Gleichungssystemen aufgeteilt.

---

```

function SOLVEHDR( $Z_{ij}, \ln \Delta t_j, N, P$ )
   $g \leftarrow \text{initG}()$ 
  while  $g$  changes do
    repeat
       $F \leftarrow \text{solveF}(F, g, Z_{ij}, \ln \Delta t_j, N, P)$ 
    until  $F$  has not changed significantly
    repeat
       $g \leftarrow \text{solveG}(F, g, Z_{ij}, \ln \Delta t_j, N, P)$ 
    until  $g$  has not changed significantly
  end while
  return [ $g, F$ ]
end function

```

---

$$F_i = \frac{\sum_{j=0}^{P-1} w^2(Z_{ij})(\mathbf{g}(Z_{ij}) - \ln \Delta t_j) \varphi'([\mathbf{g}(Z_{ij}) - F_i - \ln \Delta t_j]^2)}{\sum_{j=0}^{P-1} w^2(Z_{ij}) \varphi'([\mathbf{g}(Z_{ij}) - F_i - \ln \Delta t_j]^2)} \quad (5.43)$$

### 5.4.3 Subquadratische Bestrafungsfunktion im räumlichen Glattheitsterm von F

Die subquadratischen Bestrafungsfunktionen machen auch bei der Betrachtung des räumlichen Glattheitsterms für F Sinn. Durch die weniger starke Gewichtung von starken Ausreißern – wie sie z.B. typischerweise an Kanten in Bildern vorkommen – können Kanten im Bild besser erhalten werden.

Die Grundlage hierfür bildet der Glattheitsterm  $\Psi$  aus der Gleichung 5.28. Die darin vorkommenden quadratischen Bestrafungsterme werden nun durch die subquadratische Bestrafungsfunktion  $\varphi(s^2) = \sqrt{s^2 + \epsilon^2}$  ersetzt, was zum Term  $\tilde{\Psi}$  führt:

$$\tilde{\Psi} = \alpha \sum_{i \in A} \varphi((F_i - F_{i-1})^2) + \alpha \sum_{i=n}^{N-1} \varphi([F_i - F_{i-n}]^2) \quad (5.44)$$

Hierbei wurden die Ränder des Bildes über die folgendermaßen definierte Menge behandelt:

$$A = \{i \in [0, N - 1]\} \setminus \{i \cdot k | k \in \mathbb{N}\} \quad (5.45)$$





Das daraus entstehende Gleichungssystem in Matrix-Schreibweise ähnelt dem aus Gleichung 5.38, lediglich die Koeffizienten der Matrix  $\tilde{R}$  unterscheiden sich vom bisherigen Ansatz:

$$2H \cdot \mathbf{F} + 2\alpha\tilde{R} \cdot \mathbf{F} = 2\mathbf{b} \quad (5.48)$$

$$(H + \alpha\tilde{R}) \cdot \mathbf{F} = \mathbf{b} \quad (5.49)$$

#### 5.4.4 Subquadratische Bestrafungsfunktion im Daten- und Glattheitsterm von E

Um bei der Berechnung von  $F_i$  sowohl im Datenterm, als auch im Glattheitsterm robuste Bestrafungsfunktionen zu verwenden, müssen die Ergebnisse aus Unterabschnitt 5.4.2 und Unterabschnitt 5.4.3 kombiniert werden.

$$\begin{aligned} \tilde{\Omega} = & \underbrace{\sum_{i=0}^{N-1} \sum_{j=0}^{P-1} w^2(Z_{ij}) \cdot \varphi([\mathbf{g}(Z_{ij}) - F_i - \ln \Delta t_j]^2)}_{\text{Datenterm mit Robustheit } \tilde{\Phi}} + \lambda \underbrace{\sum_{z=Z_{min}+1}^{Z_{max}-1} [w(Z_{ij}) \cdot \mathbf{g}''(z)]^2}_{\text{Glattheitsterm für } g} \\ & + \underbrace{\alpha \sum_{i \in A} \varphi((F_i - F_{i-1})^2) + \alpha \sum_{i=n}^{N-1} \varphi((F_i - F_{i-n})^2)}_{\text{räumlicher Glattheitsterm mit Robustheit } \tilde{\Psi}} \end{aligned} \quad (5.50)$$

Die Gleichung 5.43 kann ebenfalls so umgeformt werden, dass ein LGS entsteht.

$$\begin{aligned} 2 \sum_{j=0}^{P-1} w^2(Z_{ij}) \varphi'([\mathbf{g}(Z_{ij}) - F_i - \ln \Delta t_j]^2) F_i = & \underbrace{\hspace{10em}}_{\text{Matrixeintrag } \tilde{h}_i} \\ 2 \sum_{j=0}^{P-1} w^2(Z_{ij}) (\mathbf{g}(Z_{ij}) - \ln \Delta t_j) \varphi'([\mathbf{g}(Z_{ij}) - F_i - \ln \Delta t_j]^2) & \underbrace{\hspace{10em}}_{\text{Vektoreintrag } \tilde{b}_i} \end{aligned} \quad (5.51)$$

$$2 \underbrace{\begin{pmatrix} \ddots & & \\ & \tilde{h}_i & \\ & & \ddots \end{pmatrix}}_{\text{Matrix } \tilde{H}} \cdot \underbrace{\begin{pmatrix} \vdots \\ F_i \\ \vdots \end{pmatrix}}_{\text{Vektor } \tilde{\mathbf{b}}} = 2 \underbrace{\begin{pmatrix} \vdots \\ \tilde{b}_i \\ \vdots \end{pmatrix}}_{\text{Vektor } \tilde{\mathbf{b}}} \quad (5.52)$$

$$2\tilde{H} \cdot \mathbf{F} = 2\tilde{\mathbf{b}} \quad (5.53)$$

Dieses Gleichungssystem wird nun mit der partiellen Ableitung von  $\tilde{\Psi}$  kombiniert (siehe Gleichung 5.48), was zu folgendem neuen System führt:

$$2\tilde{H} \cdot \mathbf{F} + 2\alpha\tilde{R} \cdot \mathbf{F} = 2\tilde{\mathbf{b}} \quad (5.54)$$

$$(\tilde{H} + \alpha\tilde{R}) \cdot \mathbf{F} = \tilde{\mathbf{b}} \quad (5.55)$$

Durch die Matrix  $\tilde{R}$  ist dieses Gleichungssystem jedoch erneut nichtlinear und muss wieder in eine Serie von linearen Gleichungssystemen umgeformt werden, bei denen sich nach jedem Schritt die Koeffizienten  $\varphi'$  aktualisieren. Strukturell entspricht dieses Gleichungssystem dem aus Abschnitt 5.3, da die Matrix  $\tilde{H}$  eine Diagonalmatrix mit positiven Einträgen ist. Für die Einzelschritte entstehen daher wieder positiv semidefinite und quadratische lineare Gleichungssysteme, die mit dem Successive Over-Relaxation (SOR) Verfahren (dt. Überrelaxationsverfahren) gelöst werden können.

## 5.5 Lösung der Gleichungssysteme

Grundsätzlich hat es der Algorithmus mit zwei verschiedenen Matrix-Strukturen zu tun. Bei der Berechnung von  $\mathbf{g}$  wird die strukturelle Eigenschaft der Matrix  $M$  ausgenutzt um ein schnelles Lösen mittels der LU-Zerlegung zu gewährleisten.

Bei der Erweiterung des Ansatzes um einen räumlichen Glattheitsterm (siehe Abschnitt 5.3) tritt außerdem eine positive semidefinite Matrix auf, die gut durch das SOR-Verfahren gelöst werden kann. Beide Verfahren werden hier kurz vorgestellt.

### 5.5.1 LU-Zerlegung einer Pentadiagonal-Matrix

Die Matrix  $M$  aus Gleichung 5.15 ist pentadiagonal. Das bedeutet, es sind nur die zentralen fünf Diagonal-Elemente der Matrix besetzt. Hier kommt eine besonders schnelle Variante der LU-Zerlegung zum Einsatz.

Die LU-Zerlegung ist ein Verfahren, bei dem eine quadratische Matrix  $A$  in die beiden Dreiecksmatrizen  $L$  und  $U$  zerlegt wird. Das besondere an diesem Verfahren ist, dass die nichttrivialen Elemente (Einträge ungleich null) in der Matrix  $L$  (engl. lower) sich nur in der unteren linken bzw. bei der Matrix  $U$  (engl. upper) nur in der oberen rechten Hälfte befinden. Die Diagonaleinträge von  $L$  haben alle den Wert eins.

In diesem speziellen Fall ist bekannt, dass die Matrix nur fünf besetzte Diagonalen hat. Diese Struktur ist sehr ähnlich zu der von Tridiagonalmatrizen, für die der Thomas-Algorithmus [Weso8, S. 130f] ein gängiges direktes Lösungsverfahren ist. Basierend auf diesem Verfahren wurde eine Zerlegung für pentadiagonal Matrizen entwickelt, die zu der folgenden Struktur der Matrizen  $U$  und  $L$  führt:

$$A = L \cdot U \quad (5.56)$$

$$(A_{ij}) = \begin{pmatrix} 1 & 0 & & & & \\ l_1 & 1 & 0 & & & \\ k_2 & l_2 & 1 & 0 & & \\ 0 & k_3 & l_3 & 1 & 0 & \\ & \ddots & \ddots & \ddots & \ddots & \\ & & 0 & k_n & l_n & 1 \end{pmatrix} \cdot \begin{pmatrix} m_0 & r_0 & p_0 & 0 & & \\ 0 & m_1 & r_1 & p_1 & 0 & \\ & \ddots & \ddots & \ddots & \ddots & 0 \\ & & \ddots & \ddots & \ddots & p_{n-2} \\ & & & \ddots & \ddots & r_{n-1} \\ & & & & 0 & m_n \end{pmatrix} \quad (5.57)$$

Daraus lässt sich dann der Algorithmus A.1 herleiten, der eine pentadiagonale Matrix  $A$  und einen Vektor  $\mathbf{b}$  als Eingabe hat und den Vektor  $\mathbf{x}$  zurückgibt, sodass gilt  $A \cdot \mathbf{x} = \mathbf{b}$ . Das darin enthaltene Lösen des LGS  $A \cdot \mathbf{x} = \mathbf{b}$  geschieht mittels der Vorwärts-Eliminierung und der Rückwärts-Substitution.

Eine Pivotisierung der Spalten ist nicht notwendig, da die Diagonaleinträge der Matrix bereits die betragsmäßig größten Werte einer Spalte sind. Der komplette Algorithmus ist in Pseudocode im Anhang zu finden (siehe A.1).

### 5.5.2 SOR-Algorithmus

Für die Erweiterung um einen räumlichen Glattheitsterm (siehe Abschnitt 5.4) muss außerdem noch ein weiterer Typ Matrix gelöst werden. Das dabei entstehende Gleichungssystem kann nicht so effizient mit der LU-Zerlegung gelöst werden, da die dabei entstehende Matrix nicht pentadiagonal und außerdem sehr groß ist ( $N \times N$ ).

Hierfür ist das SOR-Verfahren besser geeignet. Bei Matrizen, die quadratisch, positiv definit, symmetrischen und dünn besetzt sind, stellt dieses Verfahren eine Verbesserung gegenüber dem Gauß-Seidel-Verfahren dar. Der reelle Parameter  $\omega \in (0, 2)$  sorgt dafür, dass das Verfahren schneller konvergiert. Das Gauß-Seidel- und das SOR-Verfahren sind identisch für  $\omega = 1$ .

Die Lösung für  $x$  wird komponentenweise und iterativ nach folgender Vorschrift bestimmt:

$$x_k^{m+1} = (1 - \omega)x_k^m + \frac{\omega}{a_{kk}}(b_k - \sum_{i>k} a_{ki}x_i^m - \sum_{i<k} a_{ki}x_i^{m+1}), \quad k \in [1, n] \quad (5.58)$$

Die Implementierung des Verfahrens verwendet als Abbruchkriterium die Maximal-Norm  $r_{max}$  des Residuum-Vektors  $\mathbf{r}$  mit  $\mathbf{r} = A \cdot \mathbf{x} - \mathbf{b}$ . Diese wird nach jeder Iteration  $m$  mit  $r_{max}^m := \max_{i \in [1, n]} |r_i^m| < \delta_1$  berechnet. Falls sowohl das Residuum als auch die Differenz der Werte zweier aufeinander folgender Iterationen kleiner als die vorgegebenen Schranken sind, so terminiert das Verfahren [Weso8, S. 143]. In dieser Implementierung wurde außerdem eine maximale obere Schranke für die Anzahl der Iterationen angegeben.

# 6 Realisierung

Neben der mathematischen Ausarbeitung (siehe Kapitel 5) beschäftigt sich die vorliegende Arbeit auch mit der eigentlichen Realisierung des erarbeiteten Verfahrens. In diesem Kapitel werden die Anforderungen an die Software sowie das Vorgehen bei der Wahl der Programmiersprache beschrieben. Anschließend folgt eine Beschreibung der Software mittels eines Prototypen sowie der Architekturentwurf. Abschließend werden die Installation und Bedienung kurz erklärt.

Die entwickelte Software sowie die Dokumentation zu dieser befinden sich auf der beiliegenden CD-ROM oder können unter <https://github.com/sebastianzillessen/hdr-generator> heruntergeladen werden.

## 6.1 Anforderungen

Die Anforderungen an die Realisierung lassen sich zum einen aus der Aufgabenbeschreibung der Bachelor-Arbeit entnehmen und sind zum anderen während der Bearbeitung des theoretischen Ansatzes entstanden. Diese lassen sich in funktionale und nichtfunktionale Anforderungen gliedern [LL10, S. 369f].

### 6.1.1 Funktionale Anforderungen

**Berechnung eines HDR-Bildes:** Mit der Software soll es möglich sein, aus einer Belichtungsreihe ein HDR-Bild zu erstellen. Dazu soll das Verfahren den Algorithmus von Debevec und Malik verwenden.

**Evaluation der Daten:** Die entstehenden Daten sollen evaluiert und grafisch dargestellt werden. Dazu muss die Software in der Lage sein folgende Aufgaben zu übernehmen:

- Anpassung der Parameter für die Berechnung des HDR-Bildes
- Darstellung der Antwortkurve  $g$  als grafischen Plot
- Anzeigen des HDR-Bildes mit verschiedenen Tone-Mapping-Operatoren
- Änderung der Parameter der Tone-Mapping-Operatoren

**Mathematische Korrektheit:** Bei dieser Implementierung handelt es sich um eine interdisziplinäre Aufgabe. Die Berechnung des HDR-Bildes muss mathematisch korrekt ablaufen und soll der mathematischen Ausarbeitung (siehe Kapitel 5) entsprechen. Die Formulierung des Optimierungsproblems in effektiven Programmcode stellt deswegen eine besondere Herausforderung dar. Um die mathematische Korrektheit zu gewährleisten, sind umfassende Tests der mathematischen Klassen und Hilfsmethoden notwendig.

**Reproduzierbarkeit:** Die Berechnungen des Programms sollen reproduzierbar sein. Dadurch ist eine Evaluation der Daten unter verschiedenen Eingaben möglich.

### 6.1.2 Nichtfunktionale Anforderungen

**Portierbarkeit:** Die Implementierung der Software soll auf verschiedene Systeme portierbar sein. Dazu gehören Windows, MacOS und Linux-Derivate. Die eingesetzte Programmiersprache soll dies ohne weitere Probleme ermöglichen.

**Performanz:** An die Performanz der Implementierung wurden keine besonderen Anforderungen gestellt. Die Umsetzbarkeit und die Ausgabe des Programms stehen im Vordergrund. Aus diesem Grund wurde von einer Laufzeitanalyse des Programmes abgesehen.

**Software-Qualität:** Da diese Arbeit eine Bachelor-Arbeit der Fachrichtung *Softwaretechnik* ist, bestehen Anforderungen an die Qualität des Quellcodes. Diese beinhaltet die folgenden Aspekte:

- Automatisierte Modul-Tests (Zielwert: 90% Anweisungsüberdeckung für das Paket Maths, 80% Anweisungsüberdeckung für das Paket Solve, 90% Anweisungsüberdeckung für das Paket Model, für die Pakete View und Ctrl besteht keine Anforderung)
- Objektorientierte Programmierung (OOP)
- Die Module und public Funktionen werden gemäß dem offiziellem Javadoc Style Guide<sup>1</sup> kommentiert
- Erstellung einer Software-Dokumentation (z.B. JavaDoc)
- Gliederung des Programms in Pakete

**Grafische Benutzerschnittstelle (engl. graphical user interface) (GUI):** Die GUI soll es dem Benutzer erlauben, dem Algorithmus Bildserien und die benötigten Parameter für die Generierung eines HDR-Bildes einzugeben. Dazu gehören folgende Aspekte:

- Eingabe der Bildserie
- Eingabe der Parameter für die Bildgenerierung

<sup>1</sup><http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html>

- Eine Einweisung soll nicht erforderlich sein
- Validierung der Eingaben des Benutzers

Darüber hinaus soll der Benutzer über den Stand der Berechnung informiert werden und Zwischenergebnisse sehen können.

**Robustheit:** Bei Fehlern oder Problemen in der Software soll eine adäquate Fehlerbeschreibung geliefert werden. Das betrifft sowohl fehlerhafte Eingaben, als auch Programmfehler während der Berechnung.

## 6.2 Wahl der Programmiersprache

Für die Implementierung des Programms wurde eine Analyse verschiedener Programmiersprachen durchgeführt. Eine Übersicht der untersuchten Sprachen ist in Tabelle 6.1 zu finden. Der Vergleich zeigt, dass Go und C aufgrund der fehlenden Möglichkeit zur OOP ausscheiden. Die verbleibenden vier Sprachen Java, C++, C# und Free Pascal unterscheiden sich dann kaum noch. Lediglich die Erstellung einer GUI und die Integration von automatisierten Tests werden von den beiden zuletzt genannten Sprachen nur über zusätzliche Bibliotheken unterstützt. Java schließt nur in puncto nativem Systemzugriff und damit der Performanz schlechter ab, als die anderen Sprachen im Test. Da jedoch an die Performanz der Software (siehe Unterabschnitt 6.1.2) keine besonderen Anforderungen gestellt waren, ist dieser Punkt zu vernachlässigen.

Sprache	Java	C	C#	C++	Object Pascal <sup>a</sup>	Go
Portierbarkeit	✓	✓	(✓) <sup>b</sup>	✓	✓	✓
GUI	✓	(✓) <sup>c</sup>	✓	(✓) <sup>d</sup>	✓ <sup>e</sup>	(✓) <sup>f</sup>
OOP	✓	×	✓	✓	✓	×
Automatisierte Tests	✓	(✓) <sup>g</sup>	✓	(✓) <sup>h</sup>	(✓) <sup>i</sup>	✓
Nativer Systemzugriff	× <sup>j</sup>	✓	✓	✓	✓	✓

**Tabelle 6.1:** *Programmiersprachen im Vergleich* — ✓: *out of the box*, (✓): zusätzliche Bibliothek oder Erweiterung benötigt, ×: keine Unterstützung

<sup>a</sup>Implementierung: Free Pascal

<sup>b</sup>Plattformunabhängige Realisierung mit Mono (siehe <http://www.mono-project.com>)

<sup>c</sup>z.B. GTK+ (<http://www.gtk.org>)

<sup>d</sup>z.B. Qt (<http://qt-project.org>)

<sup>e</sup>z.B. Integration in der IDE Lazarus (<http://www.lazarus.freepascal.org>)

<sup>f</sup>z.B. go-gtk (<http://mattn.github.io/go-gtk/>)

<sup>g</sup>z.B. Check (<http://check.sourceforge.net>)

<sup>h</sup>z.B. CppUnit (<http://cppunit.sourceforge.net>)

<sup>i</sup>z.B. FPCUnit (<http://wiki.freepascal.org/fpcunit>)

<sup>j</sup>Virtuelle Maschine (Java Runtime Environment (JRE))

Aufgrund meiner in universitären Projekten erlangten Vorkenntnissen in Java und des besseren Abschneidens im Vergleich mit den anderen hier genannten Sprachen, wird die Software in Java realisiert.

### 6.3 Prototyping

Zunächst wurde unter Berücksichtigung der oben genannten funktionalen und nichtfunktionalen Anforderungen ein Prototyp der Benutzeroberfläche erstellt. Ziel dieses Prototypen ist es, die Anforderungen in der Benutzeroberfläche zu visualisieren. Die drei Ansichten des Prototyps sind in den Abbildungen 6.1 – 6.3 dargestellt.

Im Startfenster (siehe Abbildung 6.1) werden zunächst die Eingabedateien ausgewählt (1) und in einer Tabelle aufgelistet (2). Die Belichtungszeiten der Bilder werden nach Möglichkeit automatisiert aus den Dateien ausgelesen, können aber auch von Hand modifiziert werden. Die Parameter der Berechnung können über Eingabefelder (3) und Auswahlelemente modifiziert werden. Zu Testzwecken kann außerdem Rauschen auf die Eingabedateien gelegt werden. Sollte eine Eingabe nicht valide sein, so wird der Benutzer durch einen optischen Hinweis (4) darauf hingewiesen. Sobald alle Eingaben getätigt wurden, kann die Generierung des Bildes gestartet werden (5).

Während der Berechnung der Antwortkurve und des HDR-Bildes (siehe Abbildung 6.2) kann der Fortschritt (6) festgestellt werden. Außerdem wird dem Anwender die aktuell errechnete Antwortkurve grafisch dargestellt (7).

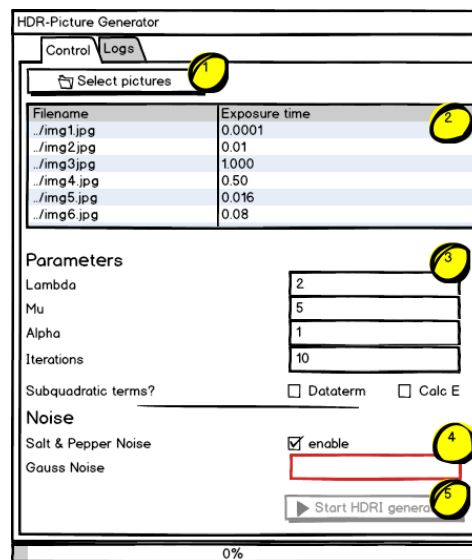


Abbildung 6.1: Prototyp der Benutzeroberfläche — Startfenster der Anwendung



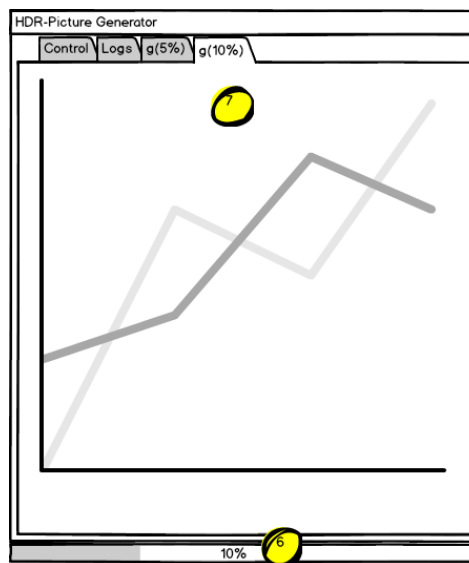


Abbildung 6.2: Prototyp der Benutzeroberfläche — Zwischenergebnisse der Antwortkurve und Fortschrittsanzeige

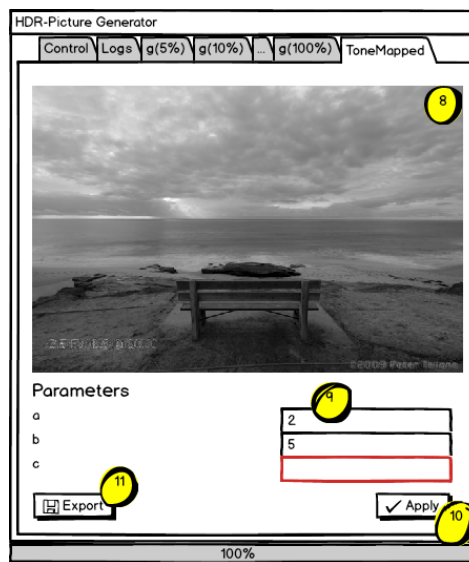


Abbildung 6.3: Prototyp der Benutzeroberfläche — Resultat mit angewandtem Tone-Mapping-Operator

Sobald das HDR-Bild erstellt wurde (siehe Abbildung 6.3), wird es dargestellt (8). Der Anwender kann die vorhandenen Parameter des Tone-Mapping-Operators verändern (9). Auch hier werden die Eingaben des Benutzers validiert und Fehler ggf. hervorgehoben. Die Veränderungen an den Parametern werden separat bestätigt (10) und das HDR-Bild kann als PNG-Bild (LDR) exportiert werden (11).

### 6.4 Architektur

Bei der Architektur der Software wurde eine Model-View-Controller (MVC) Struktur eingehalten. Dieses Muster beschreibt eine klare Trennung zwischen Datenhaltung, -verarbeitung und -repräsentation. Diese Teilung in drei Schichten sorgt dafür, dass die einzelnen Komponenten gut getestet werden können und beliebig austauschbar sind. Dadurch besteht ein hoher Grad an Erweiterbarkeit, da die Kopplung zwischen den Modulen gering ist [LL10, S. 413].

In allen Klassen wurde grundsätzlich das Prinzip des *information hiding* angewandt, wodurch eine verbesserte Erweiterbarkeit durch klare Schnittstellen gegeben ist.

**information hiding** — A software development technique in which each module's interfaces reveal as little as possible about the module's inner workings and other modules are prevented from using information about the module that is not in the module's interface specification. *IEEE Std 610.12 (1990)*

#### 6.4.1 Paketdiagramm

Die Anwendung lässt sich in fünf verschiedene Pakete trennen (siehe Abbildung 6.4). Die Zuordnung zu den drei Schichten ist in den (*Klammern*) angegeben.

**View (*View*):** Das Paket View behandelt die gesamte Darstellung der Anwendung und die grafische Auswertung der berechneten Bilder. Es enthält Tone-Mapping-Operatoren und behandelt die Benutzereingabe.

**Ctrl (*Controller*):** Dieses Paket dient zur Steuerung der Anwendung. Es werden die Eingaben verarbeitet, an die Algorithmen übergeben und an die View zurückgegeben.

**Model (*Model*):** Das Model besteht in erster Linie aus den beiden Klassen Image und HDRResult und ist damit ein Datentypmodul [LL10, S. 412]. Erstere wird als Repräsentation eines Bildes verwendet und speichert Grauwerte, Belichtungszeiten und weitere bildrelevante Informationen. Letztere dient als Kommunikationspaket zwischen dem eigentlichen Solver und der restlichen Anwendung und enthält die fertig berechnete Antwortkurve und die Radiance Map.

**Solver (*Model*):** Der Solver ist der eigentliche Kern des Programms. In diesem Paket wird die Bestimmung der Antwortkurve **g** sowie die Berechnung der Radiance Map behandelt. Hier fließen die mathematischen Erkenntnisse aus Kapitel 5 ein.

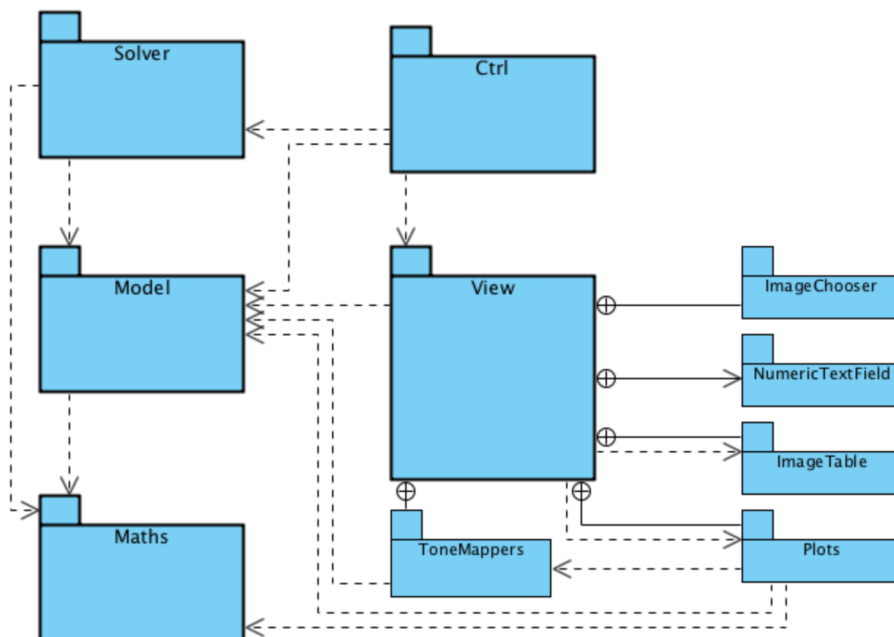
**Maths (Model):** Die Maths-Komponente ist ein Funktionales Modul [LL10, S. 412]. Es enthält alle notwendigen mathematischen Berechnungen und Repräsentationen, wie z.B. Matrix und Vector.

Nachfolgend wird nun auf die einzelnen Pakete und Klassen näher eingegangen (siehe Abbildung 6.5). Die View besteht in erster Linie aus der Klasse `GuiFrame`. Diese ist das eigentliche Fenster des Programms und übernimmt die Schnittstelle zwischen Anwender und Programm.

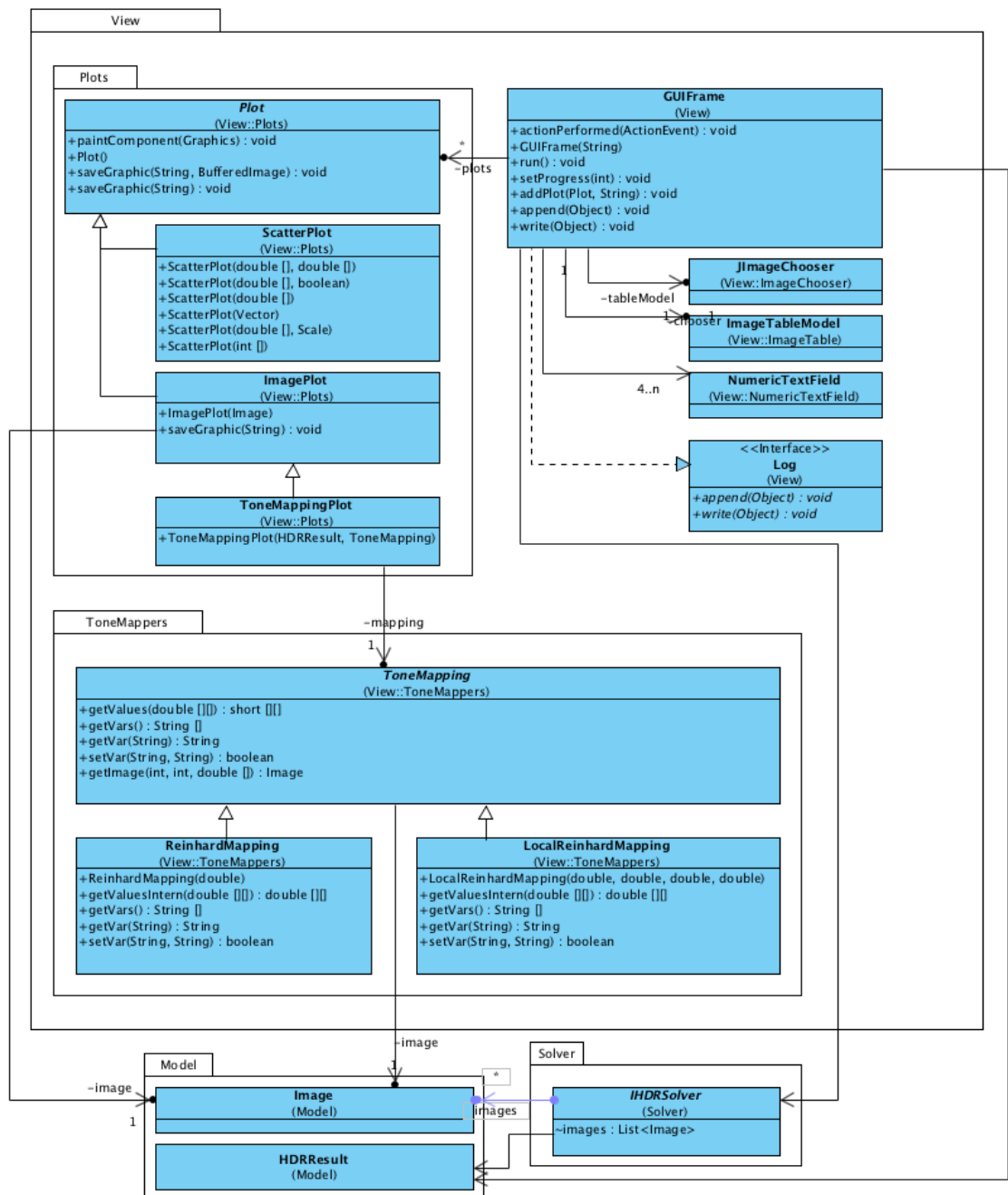
Darin enthalten sind auch die verschiedenen ToneMappers, die zur Darstellung der Radiance Map verwendet werden (Paket `Plots`).

Alle rein mathematischen Aufgaben (wie z.B. die LU-Zerlegung oder das SOR-Verfahren, siehe Abschnitt 5.5) sind im Paket `Maths` ausgelagert, da dieses Paket die gesamte mathematische Logik enthalten soll. So kann – bei Bedarf – auf eine andere Bibliothek für das Lösen der mathematischen Probleme (in aller erster Linie der Gleichungssysteme) gewechselt werden, ohne die Logik des erarbeiteten Lösungsalgorithmus zu verändern.

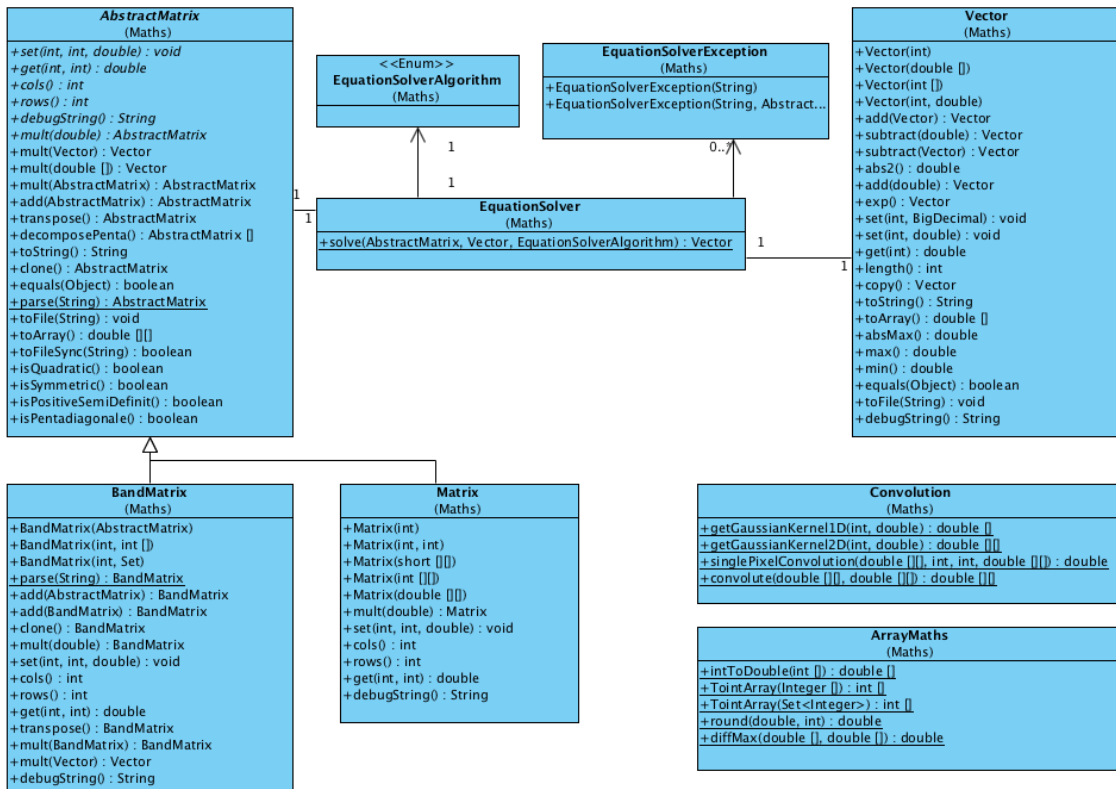
Die abstrakte Klasse `AbstractMatrix` stellt die Basisfunktionalität im Umgang mit Matrizen zur Verfügung und kennt die beiden Implementierungen `BandMatrix` und `Matrix`. Erstere ist eine spezielle Repräsentation von dünnbesetzten Diagonalmatrizen mit beliebig vielen



**Abbildung 6.4:** *Architektur der Software* — Die fünf Pakete sind Solver, View, Ctrl, Model, Maths (hier fett dargestellt). Die übrigen Pakete (ToneMappers, ImageChooser, ImageTable, Plots und NumericTextField) sind Bestandteile des Paketes View.



**Abbildung 6.5:** *View-Paket im Detail* — Das Hauptfenster GUIFrame stellt die zentrale Klasse für die Darstellung dar und importiert dazu die verschiedenen Komponenten (wie z.B. Plots und ToneMappers)



**Abbildung 6.6:** *Paket Maths* — Mathematische Bibliothek für das Lösen der Gleichungssysteme und die Repräsentationen für Matrizen und Vektoren (auf die Darstellung der Assoziationen zu anderen Klassen wurde verzichtet)

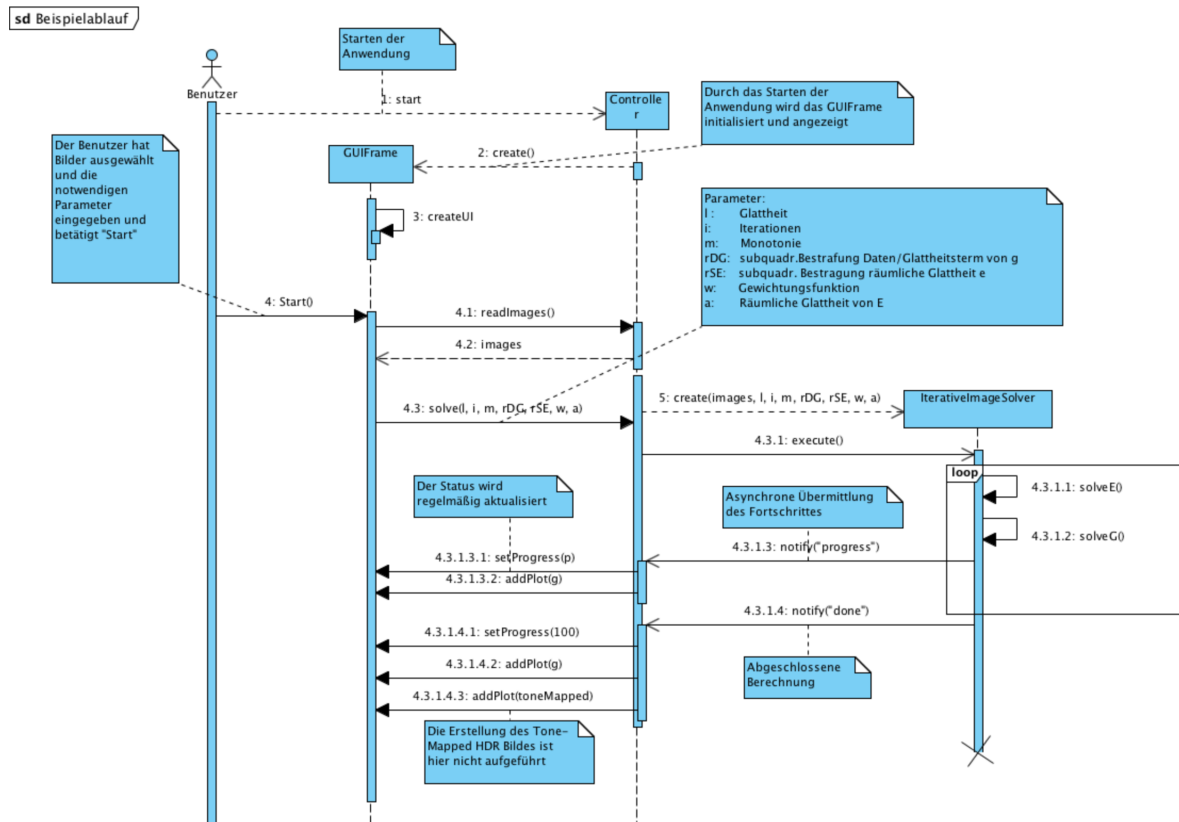
Bändern. Die hier häufig beschriebenen Pentadiagonalmatrizen werden mit dieser Implementierung dargestellt. Der Vorteil dieser ist, dass auch große, sehr dünn besetzte Matrizen abgespeichert werden können (in einem zweidimensionalen Array würden diese zu viel Speicherplatz verbrauchen). Die *Matrix* ist die zweite Implementierung der Matrizen und arbeitet intern mit einem zweidimensionalen Array.

Die Klasse *EquationSolver* (Paket *Maths*) übernimmt das Lösen eines LGS. Das SOR- und das LU-Verfahren werden unterstützt. Die beiden Hilfsklassen *Convolution* und *ArrayMaths* bieten weitere Berechnungsfunktionalitäten, die z.B. in den Tone-Mapping-Verfahren verwendet werden.

## 6.4.2 Sequenzdiagramm

Das Programm läuft sehr vielschichtig ab. Als grundsätzlicher Informationsfluss dient der Aufbau wie in *Abbildung 6.7* beschrieben. Wichtig dabei ist, dass die grafische Benutze-

## 6 Realisierung



**Abbildung 6.7:** Sequenzdiagramm — Grundsätzlicher Ablauf der Berechnung eines HDR-Bildes und der Antwortkurve (Zwischenschritte und tiefer verschachtelte Aufrufe werden nicht dargestellt)

roberfläche nicht von der Berechnungsdauer des Algorithmus blockiert wird, da so die Zwischenresultate des Algorithmus bereits analysiert werden können und der Prozess ggf. früher beendet werden kann, um Parameter zu verändern. Deshalb bestehen besonders zwischen Controller und GUIFrame asynchrone Methodenaufrufe. Das Sequenzdiagramm ist nur beispielhaft zu verstehen. Es wurden einige Aufrufe und Zwischenschritte vereinfacht oder weggelassen.

### 6.5 Externe Bibliotheken

Für die Implementierung wurden einige externe Komponenten verwendet. Diese werden hier kurz beschrieben.

**NumericTextField** Für die Eingabe der einzelnen Parameter wird die Implementierung eines numerischen Textfeldes `NumericTextField`<sup>2</sup> verwendet. Dieses verhindert ungültige Eingaben und erleichtert das Auslesen der numerischen Parameter. Diese Komponente verwendet mehrere Klassen und wurde deshalb in den nachfolgenden Diagrammen zu einem Paket gebündelt.

**metadata-extractor** Die Bibliothek `metadata-extractor`<sup>3</sup> wird verwendet um automatisiert aus Bilddateien die Belichtungszeit der Bilder auszulesen. Mit dieser wird auch das von Adobe entwickelte `xmp-core`<sup>4</sup> mit geliefert. Zusammen ermöglichen es die Bibliotheken die Belichtungszeiten der Bilder auszulesen und darzustellen.

**JImageChooser** Der verwendete modifizierte `JFileChooser`<sup>5</sup> zur Selektion der Bilddateien ist inspiriert von dem vorhandenen Tutorial bei Oracle. Er ist in der Lage eine kleine Vorschau der Bilder anzuzeigen.

## 6.6 Programmvorstellung

Für die Realisierung der GUI und der grafischen Evaluation der Ergebnisse wurde in der vorliegenden Arbeit eine Swing-Oberfläche (siehe Abbildung 6.8) implementiert, die sowohl die Eingabe der Parameter und Daten als auch die Ausgabe der verschiedenen Diagramme und Bilder ermöglichen soll. Als Vorlage wurde der Prototyp aus Abschnitt 6.3 verwendet.

Für die Auswahl der Bilddateien wurde ein Datei-Auswahl-Fenster mit Vorschau der Bilder entwickelt. Aus Gründen der Robustheit (siehe Unterabschnitt 6.1.2) können nur Bilddateien ausgewählt werden. Eine Registrierung der Bilder wird nicht unterstützt. Dies bedeutet, dass die Bilder vorab von Hand oder mit einem Algorithmus registriert werden müssen. Der Nutzer erhält diesbezüglich einen Hinweis bei der Auswahl der Bilder.

In der Tabelle werden die Bilder mit ihren Belichtungszeiten aufgeführt. Falls die Bibliothek `metadate-extractor` die Belichtungszeiten nicht auslesen konnte, müssen diese händisch eingegeben werden. Auch hierzu erhält der Benutzer ggf. einen optischen Hinweis.

Die Verwendung der Komponente `NumericTextField` (siehe Abschnitt 6.5) bei der Eingabe der Parameter stellt sicher, dass der Benutzer nur sinnvolle Daten angibt. Bei Falscheingaben ertönt ein akustisches Warnsignal und das Feld wird hervorgehoben. Die „Start“ Schaltfläche wird bei fehlenden oder fehlerhaften Angaben deaktiviert. Die Eingabemöglichkeiten werden deaktiviert sobald der Berechnungsprozess gestartet wird. Dies verhindert ungültige Benutzerinteraktionen. Detaillierte Informationen über den Fortgang der Berechnung können im LOG-Reiter der Anwendung eingesehen werden.

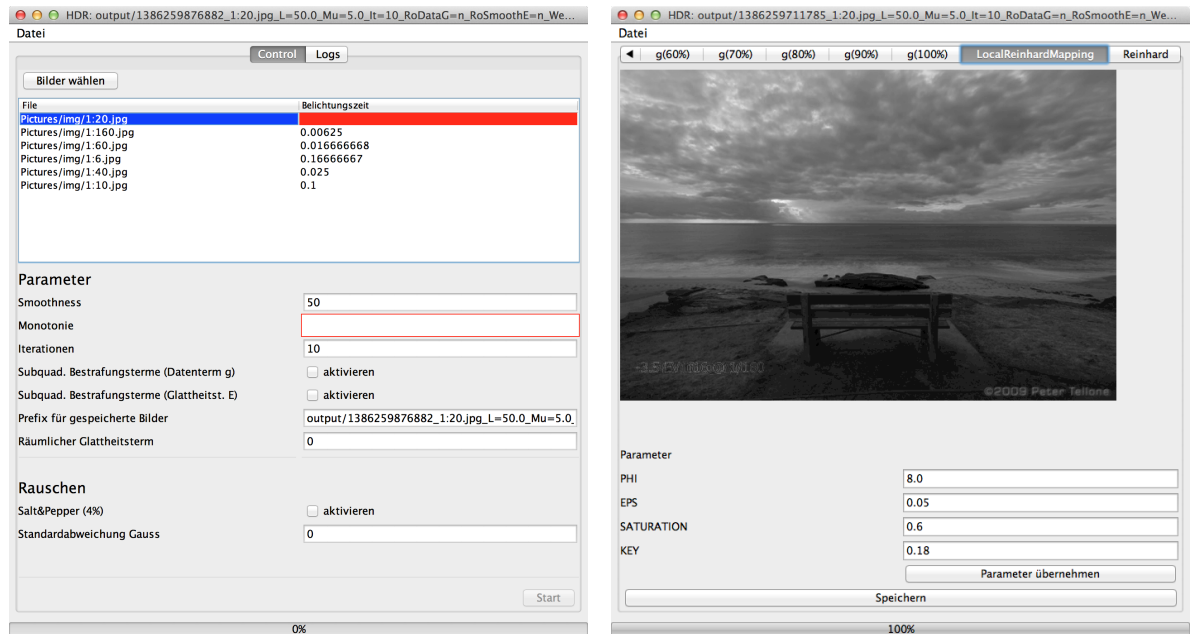
<sup>2</sup><http://www.java2s.com/Code/Java/Swing-JFC/NumericTextField.htm>

<sup>3</sup><https://code.google.com/p/metadata-extractor/>

<sup>4</sup><http://www.adobe.com/devnet/xmp.html>

<sup>5</sup><http://docs.oracle.com/javase/tutorial/uiswing/examples/components/index.html>

## 6 Realisierung



**Abbildung 6.8:** *Benutzeroberfläche* — **links:** Die Benutzereingabe für die Erzeugung der HDR-Bilder ist absichtlich schlicht und einfach gehalten. Vorlage dafür ist der Prototyp (siehe Abschnitt 6.3). **rechts:** Vorschau des Tone-Mapped Results mit der Möglichkeit der Veränderung der Variablen für diesen Tone-Mapper (lokaler Tone-Mapping-Operator von Reinhard)

### 6.6.1 Installation

Die Anwendung kann unter <https://github.com/sebastianzillessen/hdr-generator> heruntergeladen werden. Dazu wird ein installiertes JRE der Version 6 oder höher benötigt (siehe <http://www.java.com/de/download/>). Im Unterordner bin befindet sich eine ausführbare Version des Programms als `.jar` Datei.

### 6.6.2 Erste Schritte

Nach dem Öffnen der Anwendung erscheint das Hauptfenster (siehe Abbildung 6.8, links). Folgende Schritte sind notwendig um ein HDR-Bild zu erzeugen:

1. Wahl der Belichtungsserie über die Schaltfläche „Bilder wählen“.
2. Eingabe der Belichtungszeiten in der Tabelle (falls notwendig).
3. Veränderung der Parameter für die Bilderzeugung (falls gewünscht). Dazu gehören:
  - Gewichtung der Glattheitsforderung  $\lambda$  von  $g$



- Monotonie-Forderung durch die Eingabe des Gewichtes  $\mu$
- Anzahl der Haupt- und Inneniterationen (siehe Algorithmus 5.2)
- Aktivierung des räumlichen Glattheitsterms
- Aktivierung der subquadratischen Bestrafungsterme für  $\mathbf{g}$  und  $\mathbf{E}$ , sowie für den räumlichen Glattheitsterm
- Bildstörungen (Salt & Pepper Rauschen oder additives Gauss-Rauschen) auf den Eingabebildern (zu Testzwecken)
- Änderung des Präfix für den Export von Bildern

4. Bestätigung der Generierung mittels der Schaltfläche „Start“.

Dadurch wird das HDR-Bild erzeugt. Der Fortschritt wird im unteren Bereich dargestellt. Die Ausgabe (auch der Zwischenresultate) erfolgt über die verschiedenen Reiter des Fensters. Bei der Darstellung des HDR-Bilds mit den verschiedenen Tone-Mapping-Operatoren können die Parameter in der Oberfläche verändert werden (siehe Abbildung 6.8, rechts). Der Export der generierten Bilder und Kurven erfolgt über die Schaltfläche „Speichern“.



## 7 Ergebnisse und Resultate

Im nachfolgenden Kapitel werden die Einflüsse der entwickelten Erweiterungen evaluiert und mit dem Standard-Verfahren von Debevec und Malik verglichen (siehe Abbildung 7.1). Für die Experimente werden die Belichtungsserien u.a. mit additivem Gauss-Rauschen und Salt & Pepper Rauschen modifiziert, um das Verhalten der Erweiterungen unter diesen Einflüssen analysieren zu können.

Dazu werden folgende Symbole mit den angegebenen Standardwerten (falls nicht anders angegeben) verwendet:

$\lambda$  : Gewicht des Glattheitsterms für  $\mathbf{g}$ , Standard: 50

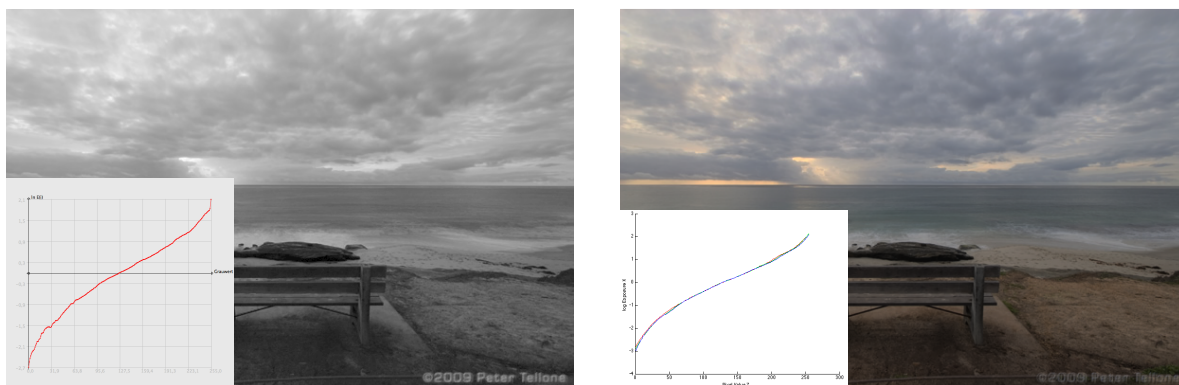
$\mu$  : Gewicht der Monotonie-Beschränkung für  $\mathbf{g}$ , Standard: 0

$\alpha$  : Gewicht der räumlichen Glattheit für  $\mathbf{E}$ , Standard: 0

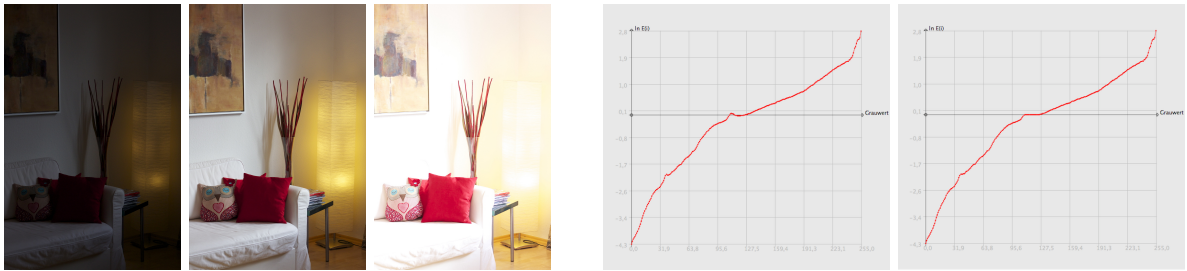
# : Anzahl der Iterationen beim iterativen Lösen, Standard: 10

$\sigma$  : Standardabweichung des additiven Gauss-Rauschen, Standard: 0

Um die Bilder einheitlich zu vergleichen, wurde bei allen Resultaten der globale Tone-Mapping-Operator von Reinhard (siehe Unterabschnitt 2.5.1, [RSSFo2]) verwendet. Außerdem wurde für nahezu alle Vergleiche die gleiche Belichtungsserie (siehe Abbildung 7.3) verwendet, welche bereits registriert ist [Tel10].



**Abbildung 7.1:** Berechnung der Antwortkurve (inkl. Resultat) — links: Mit den hier vorgestellten Erweiterungen, rechts: Herkömmliche Implementierung (hier von Mathias Eitz, siehe Abschnitt 3.1)



**Abbildung 7.2:** *Monotonie-Forderung* — Belichtungsserie (Belichtungszeiten:  $\frac{1}{8000}$ ,  $\frac{1}{640}$ ,  $\frac{1}{100}$ ), Antwortkurve ohne Monotonie-Forderung, Antwortkurve mit Monotonie-Forderung (v.l.n.r.)

### 7.1 Ergebnisse mit Monotonie-Bedingung

Im ersten durchgeführten Experiment wurde der Einfluss der Monotonie-Bedingung auf die Antwortkurve  $g$  untersucht. Bei den meisten Bildern ist diese bereits von sich aus monoton steigend, obwohl es nicht durch das Standard-Verfahren gefordert ist. Um eine Bildserie zu erhalten, bei der die Kurve diese Eigenschaft nicht bereits durch das Standard-Verfahren erhält, wurden eigene Bilder aufgenommen (siehe Abbildung 7.2). Diese Belichtungsserie liefert zunächst eine recht unförmige Antwortkurve, welche auch durch mehr Iterationen nicht weiter konvergiert. Durch die Erweiterung der Monotonie kann die Kurve entsprechend begradigt und verbessert werden. Es fällt auf, dass Überschwinger in diesem Beispiel durch die Monotonie-Bedingung quasi abgesägt werden.

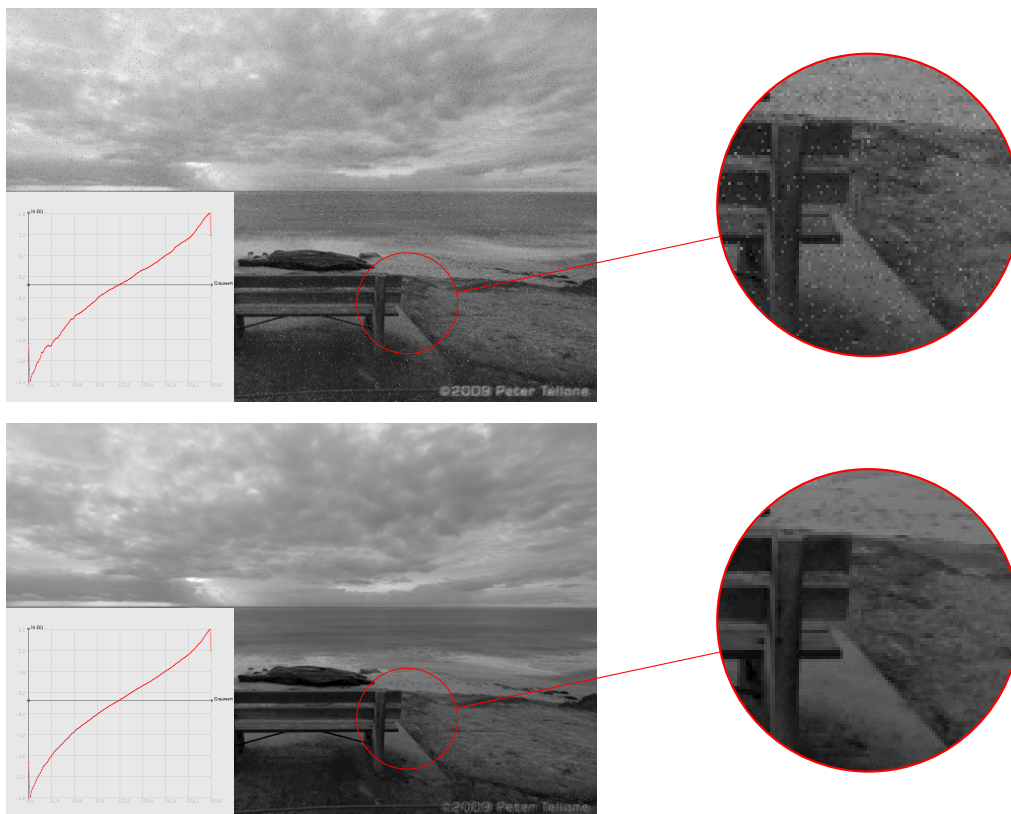
### 7.2 Ergebnisse mit räumlichem Glattheitsterm

Im zweiten Experiment wurde der Einfluss des räumlichen Glattheitsterm auf die Standard-Belichtungsserie (siehe Abbildung 7.3) untersucht.

Die Erweiterung des räumlichen Glattheitsterms sorgt dafür, dass die benachbarten Pixel bei der Berechnung der Radiance Map berücksichtigt werden. Wie in Abbildung 7.4 zu sehen, hat dies insbesondere bei Messfehlern (hier 4% Salt & Pepper Rauschen) einen enormen Vorteil gegenüber dem herkömmlichen Verfahren.



**Abbildung 7.3:** *Verwendete Belichtungsserie* — Sechs Einzelaufnahmen mit den Belichtungszeiten  $\frac{1}{6}$ ,  $\frac{1}{10}$ ,  $\frac{1}{20}$ ,  $\frac{1}{40}$ ,  $\frac{1}{60}$  und  $\frac{1}{160}$  (v.l.n.r.) [Tel10]



**Abbildung 7.4:** Räumlicher Glattheitsterm (mit 4% Salt & Pepper Rauschen) — **oben:** Antwortkurve und zugehöriges HDR Bild ohne räumlichen Glattheitsterm, **unten:** Antwortkurve mit räumlichem Glattheitsterm ( $\alpha = 1$ ) und zugehöriges HDR-Bild. Das Salt & Pepper Rauschen ist im unteren Bild quasi nicht mehr zu erkennen.

Salt & Pepper Rauschen ist eine besondere Form von Rauschen, bei dem einzelne Bildpunkte zufällig weiß oder schwarz werden. Die Prozent-Angabe in diesem Zusammenhang gibt an, welche Menge an Pixeln davon betroffen sein kann.

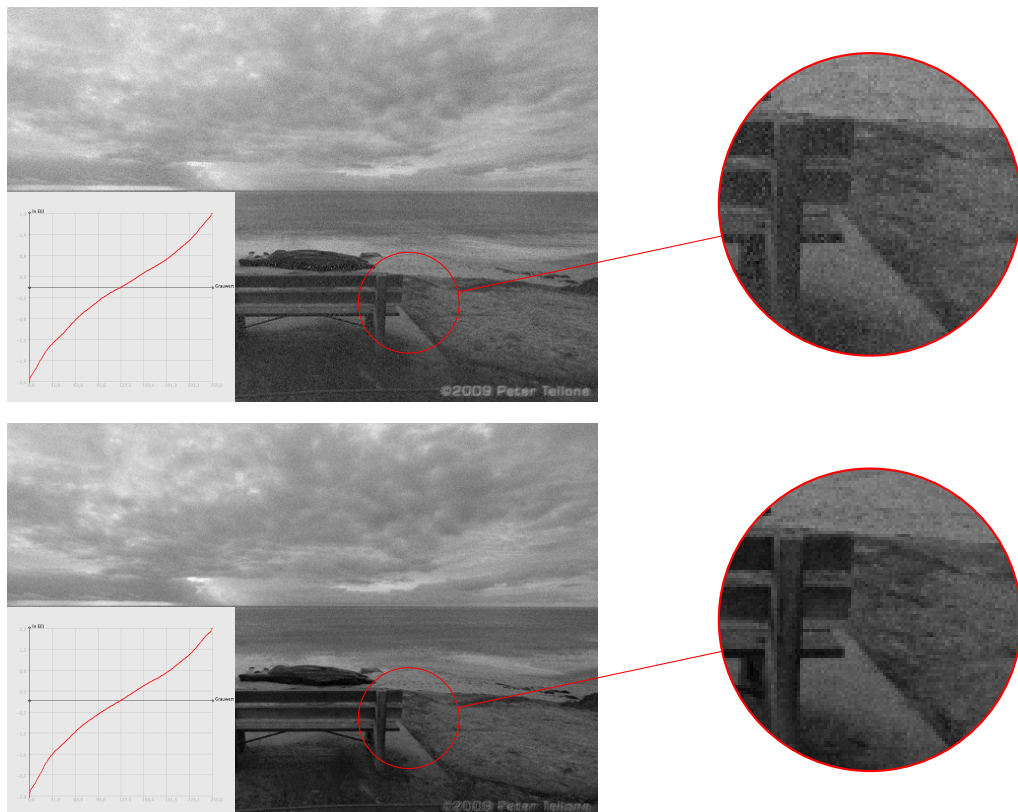
Auch durch Störungen – wie das additive Gauss-Rauschen ( $\sigma = 10$ , siehe Abbildung 7.5) – sind die Ergebnisse schärfer und das Rauschen auf den Eingangsbildern wird reduziert.

### 7.3 Ergebnisse mit subquadratischen Bestrafungstermen

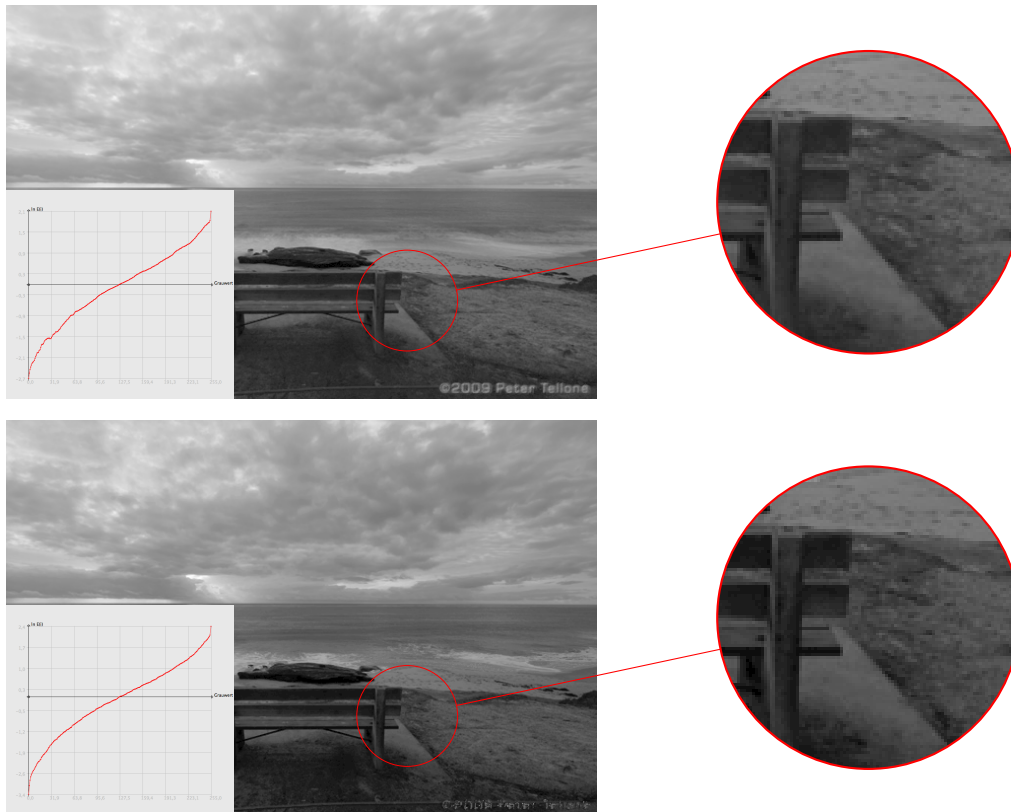
Im dritten Experiment wurde zunächst der Einfluss der subquadratischen Bestrafungsterme unter Standard-Bedingungen getestet. Unter normalen Umständen liefert das Verfahren kaum eine Verbesserung der Ausgabe (siehe Abbildung 7.6). Lediglich der Kontrast und die Konturenschärfe im HDR-Bild sind etwas verbessert.

In einem darauf aufbauendem Experiment wurde nun auch noch Rauschen (hier Salt & Pepper Rauschen, siehe Abbildung 7.7) simuliert. Dieses Störsignal konnte durch den Einsatz der subquadratischen Bestrafungsterme erfolgreich reduziert werden, sodass dieses im Ausgabebild nicht mehr zu erkennen ist. Der Effekt entsteht durch die geringere Gewichtung der Ausreißer bei der Berechnung von  $\mathbf{g}$  und  $\mathbf{E}$ .

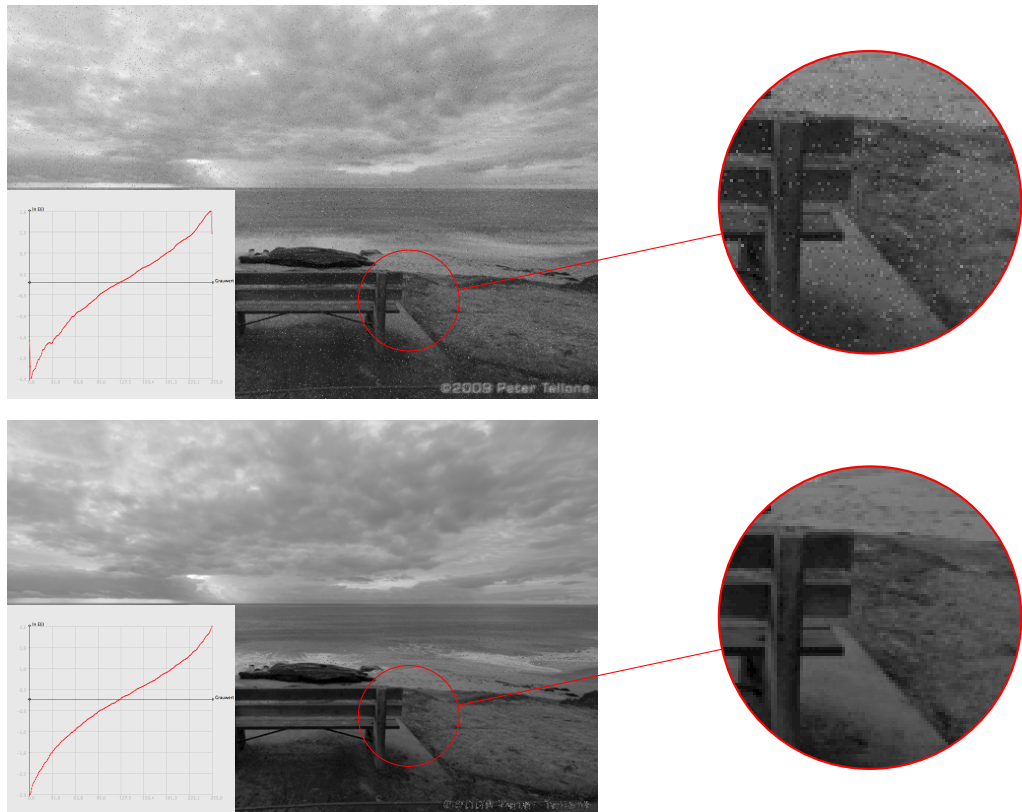
Von der Erweiterung des räumlichen Glattheitsterms um robuste Funktionen wurde eine Verbesserung der Kantenerhaltung erwartet. Diese Erwartung wurde jedoch, wie in Abbildung 7.8 zu erkennen ist, nicht erfüllt. Die Verbesserungen sind trotz hoher Gewichtung so minimal, dass sie kaum zu erkennen sind.



**Abbildung 7.5:** Räumlicher Glattheitsterm (mit additivem Gauss-Rauschen  $\sigma = 10$ ) — **oben:** Antwortkurve und zugehöriges HDR-Bild ohne räumlichen Glattheitsterm, **unten:** Antwortkurve mit räumlichem Glattheitsterm ( $\alpha = 1$ ) und zugehöriges HDR-Bild.

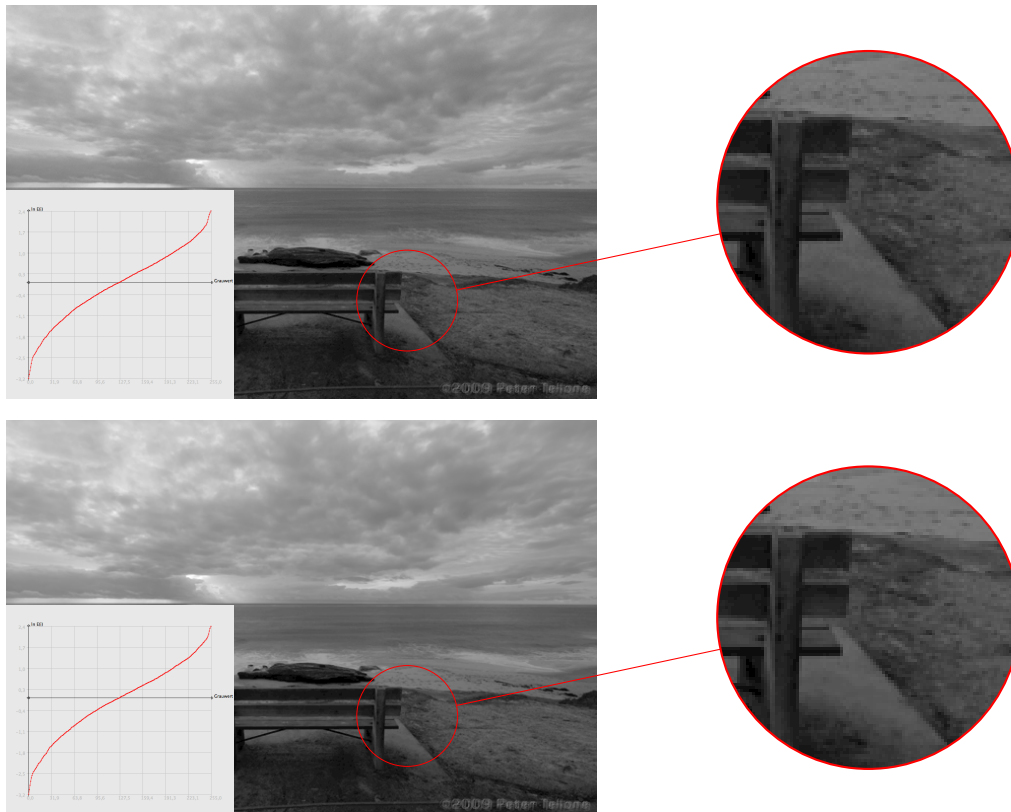


**Abbildung 7.6:** Subquadratischen Bestrafungsfunktionen — **oben:** Standardverfahren, **unten:** Subquadratischer Bestrafungsterm  $\varphi(s^2) = \sqrt{s^2 + \epsilon^2}$  für  $\mathbf{g}$  und  $\mathbf{E}$ , die Kanten sind schärfer und der Kontrast ist besser.



**Abbildung 7.7:** *Subquadratischen Bestrafungsfunktionen* — **oben:** Standardverfahren mit Salt & Pepper Rauschen, **unten:** Subquadratischer Bestrafungsterm  $\varphi(s^2) = \sqrt{s^2 + \epsilon^2}$  für **g** und **E** (das Rauschen ist quasi nicht mehr zu erkennen)





**Abbildung 7.8:** *Subquadratische Bestrafungsfunktionen (inkl. räumlicher Glattheitsterm und Salt & Pepper Rauschen,  $\alpha = 100$ )* — **oben:** Ohne subquadratische Bestrafungsfunktionen, **unten:** Mit subquadratischen Bestrafungsfunktionen



## 8 Zusammenfassung und Ausblick

Die Verwendung von HDR-Bildern ermöglicht eine dichtere Informationsdarstellung und kann dadurch in vielen Anwendungsgebieten eingesetzt werden. Besonders im Bereich der digitalen Fotografie sind HDR-Bilder Neuland für den Normalanwender, denn HDR-Kameras haben sich noch nicht auf dem Markt durchgesetzt.

Der von Debevec und Malik beschriebene Algorithmus [DM97] zur Generierung von HDR-Bildern aus einer Belichtungsserie stellt eine Alternative dar. Besonders attraktiv ist dabei, dass das Verfahren keine Kenntnisse über die kameraspezifische Antwortkurve benötigt, sondern lediglich die Belichtungszeit der einzelnen Bilder als Eingabeparameter verwendet. In modernen Digitalkameras wird diese Information im Bild gespeichert und ist somit verfügbar.

Durch die in der Arbeit beschriebenen Erweiterungen konnten Verbesserungen in der Ausgabe erzielt werden:

Die Monotonie-Forderung an die geschätzte Antwortkurve  $g$  ermöglicht eine schnellere Konvergenz des Verfahrens sowie aus physikalischer Sicht akzeptablere Resultate. Sie hat jedoch keine signifikanten Auswirkungen auf die resultierende Radiance Map.

Subquadratische Bestrafungsfunktionen machen das Verfahren besonders in Bezug auf Störungen in den Aufnahmen (wie z.B. Salt & Pepper Rauschen) robust. Dadurch können Ausreißer das resultierende Bild weniger stark beeinträchtigen und die Fehler werden in der Radiance Map reduziert.

Das Einführen des räumlichen Glattheitsterms für die Radiance Map birgt die größte Veränderung zum Algorithmus von Debevec und Malik. Durch diese Beschränkung müssen bei der Berechnung des HDR-Bildes alle Bildpunkte betrachtet werden. Dies führt zu einem Anstieg der Komplexität des Algorithmus und damit zu verlängerten Laufzeiten. Allerdings liefert diese Erweiterung (besonders bei Signalstörungen) verbesserte Ergebnisse. Durch die Kombination mit subquadratischen Bestrafungstermen kann das Verfahren robuster in Bezug auf den Erhalt von Strukturen (wie z.B. Kanten) werden.

### Ausblick

Das zu dieser Ausarbeitung entstandene Programm ist eine Machbarkeitsstudie. Es ist nicht für einen produktiven Einsatz oder die Erzeugung von ansprechenden HDR-Bildern gedacht, sondern dient der Untersuchung des Einflusses der Erweiterungen.

Die bisherige Implementierung und Ausarbeitung unterstützt nur Grauwert-Bilder. Im Artikel [DM97] wird hierzu empfohlen, die Antwortkurven der Kanäle für Rot, Grün und Blau separat zu bestimmen und diese dann bei der Wiederherstellung der Radiance Map zu verwenden. Die Erweiterungen für das Verfahren können bei dieser Strategie ebenfalls zum Einsatz kommen.

Durch die modulare Struktur der Realisierung ist z.B. eine Erweiterung durch andere Tone-Mapping-Operatoren denkbar. Kern der Arbeit war jedoch nicht der Vergleich von Tone-Mapping-Verfahren, weshalb auf die Untersuchung der Ergebnisse mit weiteren Operatoren verzichtet wurde.

# A Anhang

## A.1 LU-Zerlegung

---

**Algorithmus A.1** Lösen von  $A \cdot x = b$  mittels LU-Zerlegung (A ist pentadiagonal)

---

```
function LUDECOMPOSITION(A)
  if !ISPENTADIAGONALE(A) then
    return error
  end if
   $m_0 \leftarrow A_{0,0}$ ,  $r_0 \leftarrow A_{0,1}$ ,  $l_0 \leftarrow A_{1,0}/m_0$ ,  $m_1 \leftarrow A_{1,1} - l_1 r_0$ 
  for all  $i \in [2, n]$  do
     $p_i \leftarrow A_{i-2,i}$ ,  $k_i \leftarrow A_{i,i-2}/m_{i-2}$ 
     $r_{i-1} \leftarrow A_{i-1,i} - l_{i-1} p_{i-2}$ 
     $m_i \leftarrow A_{i,i} - k_i p_{i-2} - l_i r_{i-1}$ 
     $l_i \leftarrow (A_{i,i-1} - k_i r_{i-2})/m_{i-1}$ 
  end for
  L ← GENERATEL(l, k) // Generiert die Matrix L und U
  U ← GENERATEU(m, r, p) // nach obigem Schema (siehe Gleichung 5.57)
  return [L, U]
end function

function FORWARDELIMINATION(b,L)
   $y_0 \leftarrow b_0$ ,  $y_1 \leftarrow b_1 - L_{1,0} y_0$ 
  for  $i = 2$  to size(b) do
     $y_i \leftarrow b_i - L_{i,i-2} y_{i-2} - L_{i,i-1} y_{i-1}$ 
  end for
  return y
end function

function BACKWARDSUBSTITUTION(U,y)
   $x_{n-1} \leftarrow y_{n-1}/U_{n-1,n-2}$ 
   $x_{n-2} \leftarrow (y_{n-2} - U_{n-2,n-1} x_{n-1})/U_{n-2,n-2}$ 
   $y_1 \leftarrow b_1 - L_{1,0} y_0$ 
  for  $i = \text{size}(b) - 3$  to 0 do
     $x_i \leftarrow (U_{i,i+1} x_{i+1} - U_{i,i+2} x_{i+2} - y_i)/U_{i,i}$ 
  end for
  return x
end function

function SOLVE(A, b)
  if SIZE(A) != SIZE(b) then
    return error
  end if
  [L,U] = LUDECOMPOSITION(A);
  y ← FORWARDELIMINATION(b, L) // forward elimination  $L \cdot y = b$ 
  return BACKWARDSUBSTITUTION(U,y) // backward substitution  $U \cdot x = y$ 
end function
```

---

## A.2 MTB-Algorithmus

Dem MTB Verfahren [War03, S.9 f] wird eine Serie von  $N$  Bildern als Eingabe geliefert. Diese werden zunächst in ein Grauwert-Bild umgerechnet. Aus diesen Bildern wird der Algorithmus dann ausgehend von einem gewähltem Bild  $N - 1$  Offsets  $(x, y)$  ausgegeben, sodass die Bilder exakt übereinander gelegt werden können [RWPD10, S. 123f]. Eine Registrierung von rotierten Bildern ist somit mit diesem Verfahren nicht möglich.

Das Verfahren arbeitet dabei im Gegensatz zu vielen konventionellen Algorithmen nicht mit Kanten-Detektion im Bild um die Registrierung durchzuführen, da diese sehr anfällig auf unterschiedliche Belichtungswerte in Bildern sind. Es kommt hingegen ein Schwellwert-Verfahren auf einer Bilderpyramide zum Einsatz, dass dann mit schnellen Bit-Operationen die Verschiebung der Bilder berechnet [War03].

---

### Listing A.1 Hilfsfunktionen in C Funktion zur Berechnung der Registrierung [War03]

---

```
/** Subsample the image img by a factor of two in each dimension
 * and put the result into a newly allocated image img_ret.*/
ImageShrink2(const Image *img, Image *img_ret)

/** Allocate and compute the threshold bitmap tb and the exclusion bitmap eb for the image
    img.
 * (The threshold and tolerance to use are included in the Image struct.)*
ComputeBitmaps(const Image *img, Bitmap *tb, Bitmap *eb)

/** Shift a bitmap by (xo,yo) and put the result into the preallocated
 * bitmap bm_ret, clearing exposed border areas to zero. */
BitmapShift(const Bitmap *bm, int xo, int yo, Bitmap *bm_ret)

/** Compute the exclusive-or of bm1 and bm2 and put the result into bm_ret. */
BitmapXOR(const Bitmap *bm1, const Bitmap *bm2, Bitmap *bm_ret)

/** Compute the sum of all 1 bits in the bitmap. */
BitmapTotal(const Bitmap *bm)
```

---

---

**Listing A.2** Rekursive C Funktion zur Berechnung der notwendigen Verschiebung zwischen den Bildern um diese zu registrieren [War03]

---

```

/** Computes the shift between two images img1 and img2.**/
GetExpShift(const Image *img1, const Image *img2, int shift_bits, int shift_ret[2])
{
    int min_err;
    int cur_shift[2]; Bitmap tb1, tb2; Bitmap eb1, eb2;
    int i, j;
    if (shift_bits > 0) {
        Image sml_img1, sml_img2;
        ImageShrink2(img1, &sml_img1);
        ImageShrink2(img2, &sml_img2);
        GetExpShift(&sml_img1, &sml_img2, shift_bits-1, cur_shift); ImageFree(&sml_img1);
        ImageFree(&sml_img2);
        cur_shift[0] *= 2;
        cur_shift[1] *= 2;
    } else
        cur_shift[0] = cur_shift[1] = 0;
    ComputeBitmaps(img1, &tb1, &eb1);
    ComputeBitmaps(img2, &tb2, &eb2);
    min_err = img1->xres * img1->yres;
    for (i = -1; i <= 1; i++)
    for (j = -1; j <= 1; j++) {
        int xs = cur_shift[0] + i;
        int ys = cur_shift[1] + j;
        Bitmap shifted_tb2;
        Bitmap shifted_eb2;
        Bitmap diff_b;
        int err;
        BitmapNew(img1->xres, img1->yres, &shifted_tb2); BitmapNew(img1->xres, img1->yres,
            &shifted_eb2); BitmapNew(img1->xres, img1->yres, &diff_b); BitmapShift(&tb2, xs,
            ys, &shifted_tb2); BitmapShift(&eb2, xs, ys, &shifted_eb2); BitmapXOR(&tb1,
            &shifted_tb2, &diff_b); BitmapAND(&diff_b, &eb1, &diff_b); BitmapAND(&diff_b,
            &shifted_eb2, &diff_b);
        err = BitmapTotal(&diff_b);
        if (err < min_err) {
            shift_ret[0] = xs;
            shift_ret[1] = ys;
            min_err = err;
        }
        BitmapFree(&shifted_tb2);
        BitmapFree(&shifted_eb2);
    }
    BitmapFree(&tb1); BitmapFree(&eb1);
    BitmapFree(&tb2); BitmapFree(&eb2);
}

```

---





# Glossar und Abkürzungsverzeichnis

## **CMOS**

Complementary Metal Oxide Semiconductor. 13, 22

## **Dynamikumfang**

Kontrastverhältnis zwischen größtem und kleinstem Pixelwert im Bild. 3, 12

## **GUI**

grafische Benutzerschnittstelle (engl. graphical user interface). 46, 47, 55

## **HDR**

High Dynamic Range. 3, 7, 8, 12–18, 21–24, 26, 29, 45, 46, 48, 50, 56, 57, 61, 67

## **JRE**

Java Runtime Environment. 47, 56

## **LDR**

Low Dynamic Range. 14, 17, 21, 22, 50

## **LGS**

Lineare Gleichungssystem. 25–27, 29, 30, 32, 33, 35, 37, 53

## **MTB**

Mean Threshold Bitmap Alignment. 15, 70

## **MVC**

Model-View-Controller. 50

## **OOP**

Objektorientierte Programmierung. 46, 47

## **Pentadiagonalmatrix**

Quadratische Matrix, bei der nur die fünf zentralen Diagonalen besetzt sind (analog zu Tridiagonalmatrix). 32, 35

**Radiance Map**

Intensität des einfallenden Lichtes in den Aufnahmepunkten einer Szene. 12, 16, 25–27, 29, 50, 51, 60, 67, 68

**RAW**

Rohdatenformate von Kameraaufnahmen. 13

**Salt & Pepper Rauschen**

Das sog. Salt & Pepper Rauschen beschreibt eine Verkörnung des Bildes, bei dem Pixel auf weiß oder schwarz gesetzt sind. Dies kann durch Sensor- oder Messfehler entstehen oder durch Übertragungs- oder Abspeicherungsfehler. Salt & Pepper Rauschen kann künstlich sehr leicht produziert werden, indem einzelne Bildpunkte zufällig auf weiß oder schwarz gesetzt werden. 37, 57, 59–62, 64, 65, 67

**SOR**

Successive Over-Relaxation. 43, 44, 53

**SVD**

Singular Value Decomposition (dt. Singulärwertzerlegung). 25

**Tone-Mapping**

Tone-Mapping (dt. Dynamikkompression) steht für die Kompression des Dynamikumfangs von HDR-Bildern auf einen niederen Kontrastbereich, sodass diese mit herkömmlichen Geräten dargestellt werden können. 8, 11, 16–18, 22, 45, 49, 50, 53, 56, 57, 59, 68

# Literaturverzeichnis

- [Ado92] Adobe. Tiff specification, Revision 6.0, 1992. URL <http://partners.adobe.com/public/developer/en/tiff/TIFF6.pdf>. (Zitiert auf Seite 16)
- [BGH<sup>+</sup>06] J. Burghartz, H.-g. Graf, C. Harendt, W. Klingler, H. Richter, M. Strobel. HDR CMOS imagers and their applications. In *International Conference on Solid-State and Integrated Circuit Technology Proceedings*, S. 528–531. IEEE Press, 2006. (Zitiert auf Seite 13)
- [Blo12] C. Bloch. *The HDRI Handbook - High Dynamic Range Imaging for Photographers and CG Artists*. O'Reilly Media, 1 Auflage, 2012. (Zitiert auf den Seiten 14 und 17)
- [Bru06] A. Bruhn. *Variational Optic Flow Computation: Accurate Modelling and Efficient Numerics*. Dissertation, Department of Mathematics and Computer Science, Saarland University, 2006. (Zitiert auf Seite 27)
- [Deb98] P. Debevec. Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *ACM SIGGRAPH 1998*, S. 189–198. ACM Press/Addison-Wesley Publishing Co., 1998. (Zitiert auf Seite 13)
- [DM97] P. Debevec, J. Malik. Recovering high dynamic range radiance maps from photographs. In *ACM SIGGRAPH 1997*, S. 369–378. ACM Press/Addison-Wesley Publishing Co., 1997. (Zitiert auf den Seiten 7, 8, 23, 67 und 68)
- [FJ04] M. D. Fairchild, G. M. Johnson. The iCAM framework for image appearance, image differences, and image quality. *Journal of Electronic Imaging*, 13:126–138, 2004. (Zitiert auf Seite 12)
- [JO12] T. Jinno, M. Okuda. Multiple Exposure Fusion for High Dynamic Range Image Acquisition. *IEEE Transactions on Image Processing*, 21(1):358–365, 2012. (Zitiert auf Seite 21)
- [KYL<sup>+</sup>07] J. Kuang, H. Yamaguchi, C. Liu, G. M, M. D. Fairchild. Evaluating HDR rendering algorithms. *ACM Transactions on Applied Perception*, 4(2):9:1–9:30, 2007. (Zitiert auf den Seiten 17 und 22)
- [Lar98] G. W. Larson. LogLuv encoding for full-gamut, high-dynamic range images. *Journal of Graphics Tools*, 3(1):15–31, 1998. (Zitiert auf Seite 16)
- [LGo3] X. C. Liu, A. E. Gamal. Synthesis of high dynamic range motion blur free image from multiple captures. *IEEE Transactions on Circuits and Systems I*, 50:530–539, 2003. (Zitiert auf Seite 22)

- [LL10] J. Ludewig, H. Lichter. *Software Engeneering*. dpunkt.verlag GmbH, 2010. (Zitiert auf den Seiten 45, 50 und 51)
- [LZR12] Z. G. Li, J. H. Zheng, S. Rahardja. Detail-enhanced exposure fusion. *IEEE Transactions on Image Processing*, 21(11):4672–4676, 2012. (Zitiert auf Seite 14)
- [NMO0] S. K. Nayar, T. Mitsunaga. High dynamic range imaging: Spatially varying pixel exposures. In *IEEE Computer Vision and Pattern Recognition*, S. 472–479. IEEE Press, 2000. (Zitiert auf Seite 22)
- [RSSF02] E. Reinhard, M. Stark, P. Shirley, J. Ferwerda. Photographic Tone Reproduction for Digital Images. In *ACM SIGGRAPH 2002*, S. 267–276. ACM Press/Addison-Wesley Publishing Co., 2002. (Zitiert auf den Seiten 17 und 59)
- [RWPD10] E. Reinhard, G. Ward, S. Pattanaik, P. Debevec. *High Dynamic Range Imaging: Acquisition, Display, and Image-Based Lighting*. Morgan Kaufmann Publishers Inc., 2 Auflage, 2010. (Zitiert auf den Seiten 11, 12, 13, 15, 16 und 70)
- [Tel10] P. Tellone. How to shoot and post-process professional HDR Photos in one day, 2010. URL <http://goo.gl/cdwGdT>. (Zitiert auf den Seiten 11, 59 und 60)
- [War03] G. Ward. Fast, robust image registration for compositing high dynamic range photographs from handheld exposures. *Journal of Graphics Tools*, 8(1):17–30, 2003. (Zitiert auf den Seiten 70 und 71)
- [Weso8] T. Westermann. *Mathematik für Ingenieure - Ein anwendungsorientiertes Lehrbuch*. Springer, 6 Auflage, 2008. (Zitiert auf den Seiten 43 und 44)
- [YBMS05] A. Yoshida, V. Blanz, K. Myszkowski, H. peter Seidel. Perceptual evaluation of tone mapping operators with real-world scenes. In *Human Vision & Electronic Imaging X, SPIE*, S. 192–203. SPIE, 2005. (Zitiert auf den Seiten 17 und 22)
- [YGFT99] D. X. D. Yang, A. E. Gamal, B. Fowler, H. Tian. A 640 512 CMOS image sensor with ultrawide dynamic range floating-point pixel-level ADC. *IEEE Journal of Solid-State Circuits*, 34(12):1821–1834, 1999. (Zitiert auf Seite 12)
- [ZBW11] H. Zimmer, A. Bruhn, J. Weickert. Freehand HDR imaging of moving scenes with simultaneous resolution enhancement. *Computer Graphics Forum*, 30(2):405–414, 2011. (Zitiert auf Seite 22)

Alle URLs wurden zuletzt am 27. 11. 2013 geprüft.

## **Erklärung**

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

---

Ort, Datum, Unterschrift