

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**

**t.b.d**



# Inhaltsverzeichnis

<b>1. Einleitung</b>	<b>11</b>
1.1. Motivation . . . . .	11
1.2. Aufgabenstellung . . . . .	12
1.3. Gliederung . . . . .	12
<b>2. Grundlagen der HDR-Bilder</b>	<b>15</b>
2.1. Prinzip . . . . .	16
2.2. Anwendungsgebiet und Geschichte . . . . .	17
2.3. Bildzeugung . . . . .	18
2.4. Bildformate und -speicherung . . . . .	19
2.5. Bilddarstellung . . . . .	20
2.6. Software zur Erstellung von HDR-Bildern . . . . .	22
<b>3. Verwandte Arbeiten und Implementierungen</b>	<b>25</b>
3.1. Bekannte Implementierungen des Ansatzes von Debevec und Malik . . . . .	25
3.2. Verwandte Arbeiten . . . . .	25
<b>4. Algorithmus von Debevec und Malik [DM97]</b>	<b>27</b>
4.1. Ansatz . . . . .	27
4.2. Berechnung der Antwortkurve . . . . .	29
4.3. Konstruktion der Radiance Map . . . . .	29
4.4. Mögliche Erweiterungen des Ansatzes . . . . .	30
<b>5. Mathematische Ausarbeitung</b>	<b>33</b>
5.1. Optimierungsansatz . . . . .	33
5.2. Erweiterung um Monotonie-Eigenschaft . . . . .	38
5.3. Räumlicher Glattheitsterm . . . . .	40
5.4. Erweiterung um Robustheit . . . . .	42
5.5. Lösung der Gleichungssysteme . . . . .	47
<b>6. Realisierung</b>	<b>49</b>
6.1. Anforderungen . . . . .	49
6.2. Wahl der Programmiersprache . . . . .	51
6.3. Programmvorstellung und Benutzeroberfläche . . . . .	51
6.4. Externe Bibliotheken . . . . .	54
6.5. Architektur . . . . .	55
6.6. Herausforderungen während der Programmierung . . . . .	57

<b>7. Ergebnisse und Resultate</b>	<b>61</b>
7.1. Ergebnisse mit Monotonie-Bedingung . . . . .	62
7.2. Ergebnisse mit räumlichem Glattheitsterm . . . . .	62
7.3. Ergebnisse mit subquadratischen Bestrafungstermen . . . . .	62
<b>8. Zusammenfassung und Ausblick</b>	<b>69</b>
<b>A. Anhang</b>	<b>71</b>
A.1. Verwendete Algorithmen . . . . .	71
<b>Glossar und Abkürzungsverzeichnis</b>	<b>75</b>
<b>Literaturverzeichnis</b>	<b>77</b>

# Abbildungsverzeichnis

---

1.1.	Die Bildaufnahme Pipeline veranschaulicht, welche Prozesse durchlaufen werden bis aus einer realen Szene das digitale Bild für die Verwendung erzeugt wird (links nach rechts) [DM97, S.2] . . . . .	11
1.2.	Geschätzte Antwortkurven und zugehöriges HDR-Bild für die Bilderserie [Tel10] mit Salt & Pepper Rauschen. Das HDR-Bild wurde mittels lokalem Reinhard-Tone-Mapper (vgl. Abschnitt 2.5) erstellt. <b>links</b> : ohne robuste Bestrafungsterme. <b>rechts</b> : mit robusten Bestrafungstermen und räumlicher Glättungsforderung. . . . .	12
2.1.	Beispielhaftes High Dynamic Range (HDR)-Bild mit bereits angewandtem Tone-Mapping-Verfahren (siehe [Tel10]). . . . .	15
2.2.	Die in dieser Ausarbeitung häufig verwendete Belichtungsserie besteht aus sechs Einzelaufnahmen mit den Belichtungszeiten $1/6$ , $1/10$ , $1/20$ , $1/40$ , $1/60$ und $1/160$ (v. l.) [Tel10]. . . . .	16
2.3.	Der Erstell-Assistent von Luminance HDR. Auch hier wird intern der Algorithmus vonDebevec und Malik [DM97] verwendet. . . . .	22
2.4.	Ein mögliches Ergebnis des HDR-Bildes. Erstellt mit Luminance HDR und dem Tone-Mapping-Operator Reinhard'05 [RSD05]. Bildserie siehe [Tel10]. . . . .	23
5.1.	Einfluss der umliegenden Bildpunkte $F_i$ beim aktiviertem räumlichen Glättungsterm. Hier in 2D dargestellt . . . . .	40
5.2.	Schematischer Aufbau der Matrix $R$ für die Berechnung von $\ln E_i$ mit räumlicher Glattheit am Beispiel eines $4 \times 4$ Bildes. . . . .	42
5.3.	Einfluss der umliegenden Bildpunkte $F_i$ bei aktiviertem räumlichen Glattheitsterm mit der Erweiterung durch einen subquadratischen Bestrafungsterm . . . . .	45
6.1.	Die Benutzereingabe für die Erzeugung der HDR-Bilder ist absichtlich schlicht und einfach gehalten. . . . .	53
6.2.	Vorschau des Tone-Mapped Resultats mit der Möglichkeit der Veränderung der Variablen für diesen Tone-Mapper (globaler Tone-Mapping-Operator von Reinhard). . . . .	54
6.3.	Die Architektur der Software auf Komponentenebene. Die Trennung zwischen den einzelnen Bereichen ist hier deutlich erkennbar. . . . .	56
6.4.	Die View-Komponente im Detail. Das Hauptfenster GUIFrame stellt die Daten dar und importiert dazu die verschiedenen Komponenten. Die Plots und die ToneMappers gehören ebenfalls zu diesem Paket. . . . .	57

6.5.	Die Struktur der verschiedenen grafischen Plots. Die verschiedenen Tone-Mapping-Operatoren wurden mittels Vererbung implementiert. . . . .	58
6.6.	Die Klasse <code>Matrix</code> und ihre Spezialisierungen <code>BandMatrix</code> und <code>DefaultMatrix</code> . Diese beiden Klassen dienen zusammen mit <code>Vector</code> als ein Kernbestandteil der mathematischen Berechnungen dieses Programms. . . . .	59
6.7.	Die Klasse <code>Vector</code> dient zur Kapselung von Funktionen mit Vektoren und Matrizen. . . . .	60
6.8.	Der grundsätzliche Ablauf der Berechnung des HDR-Bildes und der Antwortkurve und der dabei beteiligten Komponenten sowie deren Interaktion. . . . .	60
7.1.	Verfahren der Berechnung der Antwortkurve und dem dazugehörigen Resultat (re.) <b>links</b> : Unser Verfahren, <b>rechts</b> : Implementierung von Mathias Eitz, siehe Abschnitt 3.1 . . . . .	61
7.2.	Einfluss der Monotonie-Forderung: Belichtungsserie (Belichtungszeiten: $1/8000, 1/640, 1/100$ ), Antwortkurve ohne Monotonie-Forderung, Antwortkurve mit Monotonie-Forderung (v.l.n.r.). . . . .	62
7.3.	Einfluss des räumlichen Glattheitsterms auf verrauschte Bilder (4% Salt & Pepper Rauschen): <b>oben</b> : Antwortkurve und zugehöriges HDR Bild ohne räumlichen Glatheitsterm, <b>unten</b> : Antwortkurve mit räumlichem Glattheitsterm ( $\alpha = 1$ ) und zugehöriges HDR-Bild. Das Salt & Pepper Rauschen ist im unteren Bild quasi nicht mehr zu erkennen. . . . .	63
7.4.	Einfluss des räumlichen Glattheitsterms auf verrauschte Bilder (additives Gauss-Rauschen mit $\sigma = 10$ ): <b>oben</b> : Antwortkurve und zugehöriges HDR Bild ohne räumlichen Glatheitsterm, <b>unten</b> : Antwortkurve mit räumlichem Glatheitsterm ( $\alpha = 1$ ) und zugehöriges HDR-Bild. . . . .	64
7.5.	Einfluss der subquadratischen Bestrafungsfunktionen: <b>oben</b> : Standardverfahren, <b>unten</b> : Subquadratischer Bestrafungsterm $\varphi(s^2) = \sqrt{s^2 + \epsilon^2}$ für $\mathbf{g}$ und $\mathbf{E}$ , die Kanten sind schärfer und der Kontrast besser . . . . .	65
7.6.	Einfluss der subquadratischen Bestrafungsfunktionen: <b>oben</b> : Standardverfahren mit Salt & Pepper Rauschen, <b>unten</b> : Subquadratischer Bestrafungsterm $\varphi(s^2) = \sqrt{s^2 + \epsilon^2}$ für $\mathbf{g}$ und $\mathbf{E}$ , das Rauschen ist quasi nicht mehr zu erkennen. . . . .	66
7.7.	Einfluss der subquadratischen Bestrafungsfunktionen auf den räumlichen Glatheitsterm bei Salt & Pepper Rauschen ( $\alpha = 100$ ): <b>oben</b> : Ohne subquadratische Bestrafungsfunktionen, <b>unten</b> : Mit subquadratischen Bestrafungsfunktionen. Die Kanten sind nur minimal besser erhalten. . . . .	67

## Tabellenverzeichnis

---

2.1.	Belichtungsstärken in verschiedenen Umgebungen [RWPDo5, S. 6] . . . . .	16
------	---	----

2.2. Verbreitete HDR-Bildformate in der Übersicht [RWPDo5, S.89]	19
6.1. Vergleich der Programmiersprachen mit den Anforderungen. ( $\checkmark$ : <i>out of the box</i> , $(\checkmark)$ : zusätzliche Bibliothek oder Erweiterung benötigt, $\times$ keine Unterstützung)	51

## Verzeichnis der Listings

---

A.1. Hilfsfunktionen in C Funktion zur Berechnung der Registrierung [Waro3]	71
A.2. Rekursive C Funktion zur Berechnung der notwendigen Verschiebung zwischen den Bildern um diese zu registrieren [Waro3]	72

## Verzeichnis der Algorithmen

---

5.1. Alternierendes Lösen nach $g(k)$ und $E_i$	33
5.2. Erweitertes alternierendes Lösen nach $g(k)$ und $\ln E_i$ mit Haupt- und Inneniterationen	44
A.1. Lösen von $A \cdot x = b$ mittels LU-Zerlegung (A ist pentadiagonal)	73



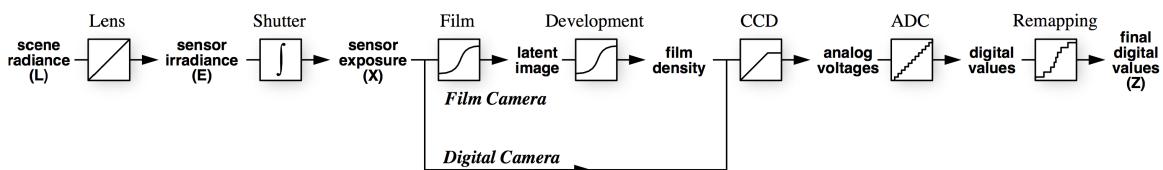
# 1. Einleitung

Die HDR-Bildgebung ist eines von vielen interessanten Problemen in dem aufstrebenden Forschungsgebiet *Computational Photography*. Ziel dieser Arbeit ist die Fusion mehrerer Bilder mit verschiedener Belichtungszeit – zu einem einzigen Bild mit deutlich vergrößertem Dynamikumfang.

## 1.1. Motivation

Während viele Arbeiten sich 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 (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, dass einen Ähnlichkeits- und einen Glattheitsterm besitzt. Während der Ähnlichkeitsterm unter Berücksichtigung der mitgeschätzten 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 macht.

Trotz der allgemeinen Formulierung hat das Verfahren von Debevec und Malik jedoch auch einige Schwachstellen. Zum einen werden weder im Daten- noch im Glattheitsterm robuste Bestrafungsfunktionen verwendet. Diese könnten den Ansatz deutlich robuster unter Fehlern messungen machen. Zum anderen werden keine Beschränkungen gefordert, die die typischerweise gewünschte Monotonie der Antwortkurve explizit erzwingen würden. Monotone Kurven können deshalb nur bei einer hinreichend großen Gewichtung der Glattheit erzielt werden.



**Abbildung 1.1.:** Die Bildaufnahme Pipeline veranschaulicht, welche Prozesse durchlaufen werden bis aus einer realen Szene das digitale Bild für die Verwendung erzeugt wird (links nach rechts) [DM97, S.2]

## 1. Einleitung

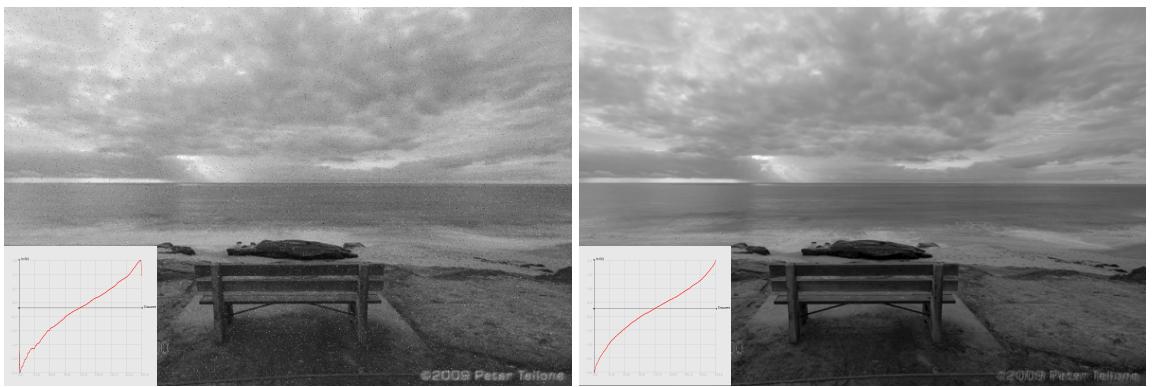
---

Schließlich ist das Verfahren auch nicht sonderlich robust gegenüber Rauschen. Dies kann insbesondere bei sehr kurz belichteten Bildern Probleme bereiten.

### 1.2. Aufgabenstellung

Ziel der Arbeit ist es zunächst, das Verfahren von Debevec und Malik [DM97] als Ausgangsverfahren zu implementieren. Dies soll in einer dafür sinnvollen und portierbaren Programmiersprache umgesetzt werden.

Diese Realisierung soll dann sukzessive um robuste Funktionen, Monotonie-Beschränkungen (siehe Abschnitt 5.2) und räumliche Glattheitsterme (siehe Abschnitt 5.3) erweitert werden.



**Abbildung 1.2.:** Geschätzte Antwortkurven und zugehöriges HDR-Bild für die Bilderserie [Tel10] mit Salt & Pepper Rauschen. Das HDR-Bild wurde mittels lokalem Reinhard-Tone-Mapper (vgl. 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

Die Arbeit ist in folgender Weise gegliedert:

**Kapitel 2 – Grundlagen der HDR-Bilder:** Hier werden die Grundlagen der HDR-Bilder vermittelt. Dabei wird auf die physikalischen, historischen und anwendungsorientierten Eigenschaften von HDR-Bildern eingegangen.

**Kapitel 3 – Verwandte Arbeiten und Implementierungen:** Anschließend werden verwandte Arbeiten zum Thema HDR vorgestellt.

**Kapitel 4 – Algorithmus vonDebevec und Malik [DM97]:** Die zu Grunde liegende Vorgehensweise von Debevec und Malik [DM97] soll in diesem Kapitel beschrieben werden. Außerdem werden die existierenden Schwachstellen des bisherigen Ansatzes dargestellt.

**Kapitel 5 – Mathematische Ausarbeitung:** Der theoretische Ansatz aus Kapitel 5 wird hier mathematisch umgesetzt und diskutiert. Außerdem werden die Erweiterungen des Algorithmus beschrieben und formal spezifiziert.

**Kapitel 6 – Realisierung:** Die Implementierung der in Kapitel 4 und Kapitel 5 beschriebenen Modelle wird hier erläutert. Darüber hinaus werden einzelne Programm-Passagen (siehe ??) genauer betrachtet.

**Kapitel 7 – Ergebnisse und Resultate:** Anschließend werden die Resultate und Einflüsse der unterschiedlichen Erweiterungen vorgestellt und diskutiert.

**Kapitel 8 – Zusammenfassung und Ausblick:** Zusammenfassung der Ergebnisse der Arbeit und Darstellung von Anknüpfungspunkten zu weiteren Arbeiten.



## 2. Grundlagen der HDR-Bilder

In den vergangenen Jahren hat die digitale Fotografie zu einem Umdenken und einer Neuschaffung von Kommunikationskanälen geführt. Im analogen Zeitalter war Fotografie in erster Linie ein autobiographisches Medium. Sie hatten u.a. im Familienfotoalbum eine Daseinsberechtigung als Gedächtnisstütze an frühere Zeiten.

Durch die Verbreitung von Digitalkameras (insbesondere auch solchen, die in Smartphones und Handys eingebaut sind) haben Fotografien aber auch eine immer größere Rolle als Kommunikationsmedium eingenommen. In diesem Zuge spielt auch die digitale Bildbearbeitung eine immer größer werdende Rolle.

Bevor auf die Grundlagen von HDR-Bildern näher eingegangen wird, bedarf es noch zunächst der Schaffung einiger Grundlagen.

Digitalbilder werden in der heutigen Zeit hauptsächlich in Form der drei Farbkanäle für 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.

Diese drei bzw. vier Kanäle werden in der Regel mittels eines Bytes repräsentiert. Damit können 16,7 Millionen verschiedene 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 zu repräsentieren (vgl. [RWPDo5, S. 1f]).



**Abbildung 2.1.:** Beispielhaftes HDR-Bild mit bereits angewandtem Tone-Mapping-Verfahren (siehe [Tel10]).

Dieses Problem wird durch die Verwendung von HDR-Bildern behoben. Ziel ist es, mehr Farben und Details in unterschiedlichen Bildbereichen sichtbar zu machen. Um dies zu

## 2. Grundlagen der HDR-Bilder

---

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 [RWPDo5, S. 6]

ermöglichen, erhöht man bei HDR-Bildern den Dynamikumfang des Bildbereiches. Dazu bricht man die Beschränkung auf den Byte-Bereich auf und erlaubt Fließkomma-Werte im Bildbereich.

### 2.1. Prinzip

Das menschliche Auge kann in einer täglichen Szene einen Dynamikumfang im Bereich von 1:10.000 (vgl. [FJo04]) wahrnehmen. Dies liegt weit über den herkömmlichen Werten eines normalen Kamera-Sensors. In der Tabelle 2.1 können verschiedenen Dynamikumfänge (und die damit zusammenhängende Beleuchtungsstärke) entnommen werden. Um wie gewünscht mit HDR-Bildern einen höheren Dynamikumfang darstellen zu können, müssen daher mehr Informationen als über den herkömmlichen Weg beschafft werden. 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 die höhere Dichte der Bildinformationen (z.B. die großen Helligkeitsunterschiede) aufzunehmen (vgl. [YGFT99]).

Der hier verwendete Begriff „Dynamikumfang“ beschreibt das Verhältnis zwischen hellstem und dunkelstem Pixel im Bild. Um Ausreißer weniger zu berücksichtigen und die Messung robuster zu machen, werden hierbei auch Quantile verwendet, die dafür sorgen sollen, dass Rauschen nicht ins Gewicht fällt. Bei Bildschirmen hingegen wird unter dem Dynamikumfang das Verhältnis zwischen der maximalen und minimalen Leuchtkraft verstanden (vgl. [RWPDo5, S. 4]).



**Abbildung 2.2.:** Die in dieser Ausarbeitung häufig verwendete Belichtungsserie besteht aus sechs Einzelaufnahmen mit den Belichtungszeiten  $1/6$ ,  $1/10$ ,  $1/20$ ,  $1/40$ ,  $1/60$  und  $1/160$  (v. l.) [Tel10].

## 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 (vgl. [RWPDo5, S. 87f]).

**Digitale Fotografie:** Die verschiedenen Kamera-Hersteller gehen bereits immer mehr in Richtung der sog. „aufnahmeabhängigen Daten“. In diesen sind bereits häufig mehr Bildinformationen enthalten. Dabei handelt es sich bei verschiedenen Herstellern in der Regel jedoch auch um verschiedene Rohdatenformate (RAW), die meist 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:** Eine der ersten Anwendungen waren vermutlich die ersten Render-Engines von Visualisierungen (Computer-Spiele, medizinische Visualisierungen und Simulationen, etc.). 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 Bildbearbeitungs-Anwendungen bieten mittlerweile in der Regel auch die Bearbeitung und Generierung von HDR-Bildern an. Als Beispiele sind hier Adobe Photoshop<sup>1</sup>, Photogenics<sup>2</sup> und Photomatix<sup>3</sup> genannt.

**Medizin:** In der Endoskopie besteht ein hoher Bedarf an immer höher auflösenden Complementary Metal Oxide Semiconductor (CMOS) Bildsensoren. Diese können immer bessere Aufnahmen aus dem Inneren des Körpers liefern und helfen damit in der Medizin große Fortschritte machen zu können. Solche Sensoren können bereits in der Größe eines Streichholzkopfes einen Dynamikumfang von 179 dB erreichen (vgl. [BGH<sup>+o6</sup>]).

**Virtual Reality:** Bei Anwendungen, bei denen sich der Benutzer in einem virtuellen Raum bewegt, wird die Wahrnehmung zunehmend wichtig. Auch hier spielen deshalb hohe Dynamikumfänge eine besondere Rolle. Außerdem ist es besonders in diesem Bereich wichtig, gute Kompressions-Algorithmen für HDR-Bilder zu entwickeln, um eine schnelle Übertragung dieser zu gewährleisten. Auch bei dem Platzieren von synthetischen Objekten in realen Szenen (vgl. [Debo8]) können HDR-Bilder eingesetzt werden, um dem Betrachter eine noch „realere“ Szene zu suggestieren.

<sup>1</sup><http://adobe.com/photoshop>

<sup>2</sup><http://www.cinepaint.org>

<sup>3</sup><http://www.hdrsoft.com/download.html>

## 2.3. Bilderzeugung

Für die Erstellung von HDR-Bildern gibt es unterschiedliche Möglichkeiten. Dabei muss man jedoch zwischen echten HDR-Bildern und „Pseudo-HDR“ Bildern unterscheiden. Im Nachfolgenden werden die verschiedenen Verfahren kurz beschrieben. Der Fokus liegt jedoch auf dem letzten Verfahren, der HDR-Bildgenerierung aus einer Belichtungsreihe.

### 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 Low Dynamic Range (LDR)-Bildern zu generieren, ohne dabei ein HDR-Bild zu erzeugen (vgl. [LZR12]). Die Bilder der Belichtungsreihe werden dazu einfach fusioniert. Diese Technologie wird hier jedoch nicht behandelt.

### 2.3.2. HDR-Kameras

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

### 2.3.3. HDR-Bildgenerierung aus einer Belichtungsreihe

Um ein HDR-Bild aus einer Belichtungsreihe zu erzeugen, braucht man zunächst die Grundlage für das Bild. Dazu sind in der Regel mehr Informationen notwendig als eine einzelne Aufnahme liefern kann. Deshalb werden mehrere Bilder der selben Szene mit unterschiedlichen Belichtungszeiten aufgenommen. Ziel der Algorithmen ist es, anschließend aus diesen Bildern ein HDR-Bild zu erzeugen.

Um die Bilder später weiter zu verarbeiten, müssen diese jedoch zunächst registriert werden. Dies ist aufgrund der verschiedenen Belichtungswerte der Aufnahmen nicht über Kanten detektionsverfahren möglich, da diese Merkmale unter den unterschiedlichen Belichtungen sehr stark variieren können.

Ein performanter Ansatz um Bilder zu registrieren ist der Mean Threshold Bitmap Alignment (MTB) Ansatz (siehe Unterabschnitt A.1.1). Auf eine Implementierung dieser Verfahren wurde verzichtet, da sie nicht relevant für die Zielsetzung sind.

## 2.4. Bildformate und -speicherung

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

Tabelle 2.2.: Verbreitete HDR-Bildformate in der Übersicht [RWPDo5, S.89]

## 2.4. Bildformate und -speicherung

Für die Abspeicherung der HDR-Bilder werden in Tabelle 2.2 die drei gängigsten Formate mit den dazu gängigen Codierungen verglichen (vgl. [RWPDo5]). Die Speicherung der erzeugten Daten war kein zentraler Bestandteil dieser Arbeit und wird vom Programm auch nicht unterstützt. Dennoch sollen hier die bekanntesten Kodierungen zur Abspeicherung vorgestellt werden.

### 2.4.1. RGBE – Das .hdr Format

Dieses Format wurde ursprünglich unter den Dateiendungen *.hdr* und *.pic* eingeführt. Abgesehen von den Metadaten (wie z.B. Bildgröße, Ausrichtung, notwendigen Angaben zur verwendeten Kodierung, etc.) werden die Bildpunkte mit 32-Bit dargestellt. Diese umfassen die Kanäle für Rot, Grün und Blau sowie einen Exponenten, was zu einer Vergrößerung des Dynamikbereiches führt [RWPDo5, S. 92].

### 2.4.2. TIFF – Gleitkomma Codierung

Das Format *.tif[f]* enthält eine 32-Bit Kodierung pro Komponente (also 96-Bit für einen Bildpunkt). Dabei werden die Bildpunkte mittels Fließkommazahlen dargestellt [Ado92].

## 2. Grundlagen der HDR-Bilder

---

Dieser Standard unterstützt bereits eine sehr hohe Genauigkeit. Dazu benötigt dieses Dateiformat im Vergleich zu anderen jedoch auch am meisten Speicherplatz. Allerdings lassen sich nahezu verlustfreie Abspeicherungen von HDR-Bildern erreichen. Im Standard von 1992 wurde auf jede Form der Komprimierung verzichtet [RWPDo5, 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<sup>4</sup> wurde 2002 veröffentlicht und basiert ebenfalls auf der Speicherung von Fließkommazahlen. Dabei ist es jedoch auch möglich die Fließkommazahlen nur mit 16 Bit (Hälfte der normalen Anzahl) abzuspeichern: 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 die Kompression quasi verlustfrei durchführbar (vgl. [RWPDo5, S. 97f]).

## 2.5. Bilddarstellung

Für die Darstellung der Radiance Maps gibt es vereinzelte spezielle Hardware, die in der Lage sind den erweiterten Dynamikumfang darzustellen. Sehr viel häufiger kommen jedoch sog. Tone-Mapping (dt.: Dynamikkompressions) Verfahren zum Einsatz. Diese stellen ein Bild mit erhöhtem Dynamikumfang durch eine andere Skalierung des Bildbereichs auf handelsüblichen Monitoren oder als herkömmliche Bilddateien dar.

Der Kerngedanke beim Tone-Mapping besteht darin, einen geeigneten Weg für die Zuordnung von Bildpunkten aus dem HDR-Bild in das LDR-Bild zu finden (vgl. [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. Die Veränderungen können auf den verschiedenen Farbkanälen unterschiedlich sein. Auch die Berechnung der modifizierten Kurve kann sich aufgrund des Bildes ändern (vgl. [Blo12, S. 146]).

<sup>4</sup>[www.openexr.com](http://www.openexr.com)

Es bestehen viele verschiedene Ansätze für globale Tone-Mapping-Operatoren, die unterschiedliche Vor- und Nachteile aufweisen können. In dieser Arbeit wurde lediglich der Tone-Mapping-Operator von Reinhard et al. [RSSF02] implementiert und verwendet. Dies ist 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).  $\alpha$  bestimmt die Lage des mittleren Grauwertes und hat in der Regel den Wert 0.18. Daraus lässt sich dann der einfache 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

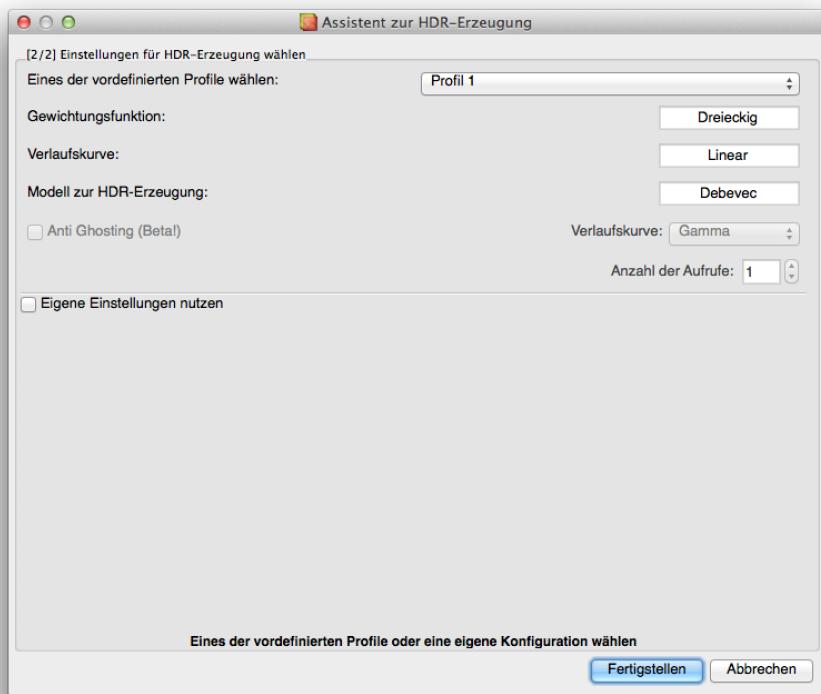
Lokale Tone-Mapping-Operatoren können bei der Zuordnung eines Wertes aus dem HDR-in das LDR-Bild auch die lokale Umgebung eines jeden Bildpunktes berücksichtigen. Damit erreichen Sie besonders in sehr dynamischen Bildern bessere Ergebnisse und können mehr Details in Bildern hervorheben. Auch hier gibt es eine Vielzahl verschiedener Operatoren, die alle ihre Vor- und Nachteile haben. Da keine Analyse der Tone-Mapping-Operatoren im Fokus der Arbeit stand, wurde hier ebenfalls ein Operator gewählt. Die Wahl dieses Operators wurde nach ersten Recherchen getroffen, da dieser in verschiedenen Veröffentlichungen [YBMS05, KYL<sup>+</sup>07] gut abgeschnitten hat.

Reinhard et al. [RSSF02] stellen in ihrer Ausarbeitung auch einen weitaus komplexeren Operator vor, der die lokalen Eigenschaften des Bildes mit analysiert und entsprechend den Operator anpasst. 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 dies bei der Entwicklung des Filmmaterials differenziert zu behandeln. Die Wahl der einzelnen Bereiche geschieht im technischen Ansatz durch die Berechnung des lokalen Kontrastes im Bild. Die Reichweite des Einflusses der umliegenden Bildpunkte wird über diese Bereiche gesteuert.

Auf eine weitere Beschreibung des Verfahrens wird hier aus Gründen der Komplexität verzichtet.

## 2.6. Software zur Erstellung von HDR-Bildern

Herkömmlich Programme zur Bildbearbeitung (z.B. Photoshop<sup>5</sup> oder GIMP<sup>6</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 trümpfen mit hohem Funktionsumfang, vielfältigen Einstellungsvarianten und der Möglichkeit der weiteren Bearbeitung auf.



**Abbildung 2.3.:** Der Erstell-Assistent von Luminance HDR. Auch hier wird intern der Algorithmus von Debevec und Malik [DM97] verwendet.

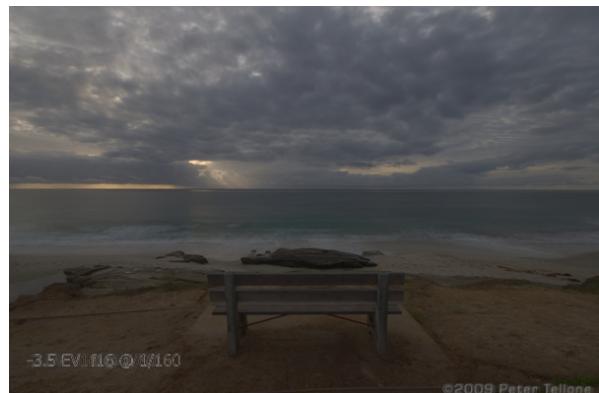
Darüber hinaus gibt es verschiedene Programme, die speziell auf die Erzeugung von HDR-Bildern spezialisiert sind (z.B. Photomatix<sup>7</sup> oder Luminance HDR<sup>8</sup>). Der Funktionsumfang dieser Programme ist verhältnismäßig klein, führt jedoch auch Laien schnell zum Ziel, da

<sup>5</sup><http://www.adobe.com/de/products/photoshop.html>

<sup>6</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>7</sup><http://www.hdrsoft.com/de/>, kostenpflichtig

<sup>8</sup><http://qtpfsgui.sourceforge.net>, Freeware



**Abbildung 2.4.:** Ein mögliches Ergebnis des HDR-Bildes. Erstellt mit Luminance HDR und dem Tone-Mapping-Operator Reinhard'05 [RSD05]. Bildserie siehe [Tel10].

(wie z.B. bei Luminance HDR, siehe Abbildung 2.4) interaktive Assistenten den Benutzer bei der Erstellung anleiten.

Als weitere Beispiele für HDR-Software seien hier außerdem Dynamic-Photo HDR<sup>9</sup> und HDR Darkroom<sup>10</sup> genannt. Diese haben sehr viele Tone-Mapping-Operatoren implementiert und können sowohl realistische als auch sehr verfremdete HDR-Bilder generieren.

Ein wirklich einfaches Programm ist Pictureonaut<sup>11</sup>. Die Anzahl und der Funktionsumfang der implementierten Tone-Mapping-Operatoren ist limitiert, jedoch liefert das Programm recht rasch realitätsgetreue Bilder.

<sup>9</sup><http://www.mediachance.com/hdri/index.html>, kostenpflichtig

<sup>10</sup><http://www.everimaging.com>, kostenpflichtig

<sup>11</sup><http://www.hdrlabs.com/pictureonaut/>, Freeware



## 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 Belichtungsreihe 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. Am ursprünglichen Artikel von Debevec und Malik ist bereits eine MATLAB-Version des Algorithmus angefügt. 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 von Bildpunkten (siehe Unterabschnitt 4.4.2) geschieht in dieser Implementierung in einer Art Rasterung der Bilder und berücksichtigt keine der Forderungen von Debevec und Malik (siehe Unterabschnitt 4.4.2).

Auch in der beschriebenen Software zur Erstellung von HDR-Bildern (siehe Abschnitt 2.6) wird z.T. dieser Ansatz verwendet.

### 3.2. Verwandte Arbeiten

In den letzten Jahren haben die veröffentlichten Arbeiten zur Generierung, Darstellung und Verarbeitung von HDR-Bildern stetig zugenommen. Deswegen hat die nachfolgende Auflistung keinen Anspruch auf Vollständigkeit und dient lediglich einer groben Übersicht.

Nayar et al. [NMoo] 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.

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

### 3. Verwandte Arbeiten und Implementierungen

---

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

Auch die Tone-Mapping-Operatoren werden ständig untersucht und verbessert. So verglichen 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 verwendeten Operatoren durchweg in allen Szenen besser abgeschnitten hat als die Mitgetesteten. Dies führt zur Schlussfolgerung, dass eine große Korrelation zwischen Szene und Tone-Mapping-Verfahren vorliegt.

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 Haupterkenntnisse dieser Studie war, dass die lokalen Operatoren die Bilddetails besser beibehalten und die globalen Operatoren den Kontrast besser darstellen 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 behandelt um Rauschen zu reduzieren.

Im Artikel von Zimmer et al. [ZBW11] wird ebenfalls ein Verfahren zur Reduktion von Bewegungsunschärfe 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 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 dadurch ein HDR-Bild zu erzeugen.

## 4. Algorithmus vonDebevec und Malik [DM97]

Diese Arbeit behandelt im Kern den Ansatz von Paul E. Debevec und Jitendra Malik [DM97]. Obwohl der Artikel bereits relativ alt ist (Verfassung 1997), wird das Verfahren noch immer in vielen Anwendungen benutzt (siehe Abschnitt 3.1). Dessen Kerngedanke 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 digital weiterverwertbare Daten 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 kalibriert werden. Beim Ansatz von Debevec und Malik hingegen wird diese Kamera-Antwortfunktion während der Generierung des HDR-Bildes aus der Belichtungsserie errechnet. Damit bietet er die Möglichkeit, 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 unterschiedlichen Belichtungen in der Bildserie

$N$ : 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 aus Gründen der Vereinfachung mit 0 belegt)

#### 4. Algorithmus von Debevec und Malik [DM97]

---

$Z_{max}$ : Maximaler Grauwert  $Z_{max} = \max\{Z_{ij}\} \forall i, j$  (wird aus Gründen der 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  $j \in [0, P - 1]$

$f(X)$ :  $f$  sei die nichtlineare Funktion, welche aus einer Belichtung  $X$  in einem Pixel einen Grauwertbild  $Z$  erzeugt mit  $f(X) = Z$

$\mathbf{g}(z)$ : Vektor mit 256 Einträgen und damit diskret (abuse of notation)

$\mathbf{g}'(z), \mathbf{g}''(z)$ : Approximation der ersten bzw. zweiten Ableitung der diskret definierten Funktion  $\mathbf{g}(z)$  ( $\mathbf{g}'(z) = \mathbf{g}(z) - \mathbf{g}(z - 1)$ ,  $\mathbf{g}''(z) = \mathbf{g}(z - 1) - 2\mathbf{g}(z) + \mathbf{g}(z + 1)$ )

#### 4.1.2. Herleitung

Da aus physikalischer Sicht angenommen werden kann, dass  $f$  monoton steigend 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.

Mithilfe dieses Rahmens lassen sich folgende Zusammenhänge darstellen:

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

Das obige Gleichungssystem hat die Unbekannten  $\mathbf{g}(z)$  und  $E$ . Um das Gesamtsystem zu lösen und das HDR-Bild zu erzeugen lässt sich folgendes Energiefunktional aufstellen, welches minimiert werden muss:

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

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 von Gleichung 4.1 kann  $\mathbf{g}$  nicht konkret bestimmt werden. Durch die Minimierung bleibt ein Skalierungsfaktor  $\alpha$  unbekannt. Dies ist daran ersichtlich, dass ein Ersetzen von  $\ln E_i$  durch  $\ln E_i + \alpha$  und  $\mathbf{g}$  durch  $\mathbf{g} + \alpha$  keine Änderung in Gleichung 4.1 hervorrufen würde. Um jedoch klare Ergebnisse für die Antwortkurven zu erhalten wird eine weitere Bedingung für  $\mathbf{g}$  dem Linearen Gleichungssystem (LGS) hinzugefügt. Diese besagt, dass der mittlere Grauwert  $Z_{mid} = \frac{1}{2} \cdot (Z_{min} + Z_{max})$  auch eine einheitliche Beleuchtung erhalten soll:  $\mathbf{g}(Z_{mid}) \stackrel{!}{=} 0$

## 4.2. Berechnung der Antwortkurve

Aus dem Energiefunktional (siehe Gleichung 4.1) lassen sich durch partielle Ableiten nach  $E_i$  und  $\mathbf{g}(k) \forall k \in [Z_{min}, Z_{max}]$  mehrere Gleichungen erstellen. Debevec und Malik schlagen vor dieses überbestimmte 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 zentrale Approximation 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 bei  $Z_{mid}$  ( $\mathbf{g}(Z_{mid}) = 0$ ) verwendet.

## 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 nach  $E_i$  umgestellt werden kann.

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

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

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  $\ln E_i$  alle Bilder der Belichtungsserie zu verwenden und diese gewichtet zu mitteln (siehe Gleichung 4.4).

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

## 4.4. Mögliche Erweiterungen des Ansatzes

Der grundlegende Ansatz von Debevec und Malik (Gleichung 4.1) hat einige Schwachstellen. Diese werden zum Teil bereits durch die Autoren des Artikels (vgl. [DM97]) angesprochen und werden hier der Vollständigkeit halber aufgelistet.

### 4.4.1. Gewichtungsfunktion

Da  $\mathbf{g}$  typischerweise sehr steil in der Nähe von  $Z_{min}$  und  $Z_{max}$  sein wird, macht es Sinn diese Randbezirke bei der Berechnung von  $\mathbf{g}$  weniger stark zu gewichten. Aus diesem Grund wird eine Gewichtungsfunktion  $w(z)$  als Dreiecks-Funktion eingeführt (siehe Gleichung 4.5). Durch diese werden die Terme des Energiefunktionalen in der Mitte stärker gewichtet und die steilen äußeren Bereiche der Kurve  $\mathbf{g}$  weniger. Diese Gewichtungsfunktion wird außerdem auch bei der Rekonstruktion der Radiance Map verwendet, um den Einfluss der Bildpunkte über die gesamte Belichtungsreihe zu mitteln. Diese Veränderung wird in das Energiefunktional (siehe Gleichung 4.6) eingearbeitet.

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

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

### 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  $N \cdot (P - 1) > (Z_{min} - Z_{max})$  gilt. 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. Außerdem werden die  $E_i$  bei diesem Verfahren erst anschließend berechnet.

Diese Selektion der Referenzpunkte aus den Belichtungsreihen wird von Debevec und Malik noch händisch durchgeführt. Bei 11 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 das 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.

#### 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 im Bezug auf Konstanzannahmen nicht robust. Eine typische Erweiterung ergibt sich durch den Einsatz von nichtlinearen Bestrafungsfunktionen (vgl. [Bruo6, S. 9f, S. 87f]). Diese haben den Vorteil, dass sie Ausreißer in der Eingabe (wie z.B. Messfehler oder Rauschen) bei der Minimierung abschwächen und diese somit das Ergebnis weniger stark beeinflussen. Hier wird eine sog. subquadratische Bestrafungsfunktion (siehe Gleichung 4.7) zusammen mit ihrer Ableitung eingesetzt.

$$\varphi(s^2) = \sqrt{s^2 + \epsilon^2} \quad \varphi'(s^2) = \frac{1}{2\sqrt{s^2 + \epsilon^2}} \quad (4.7)$$

#### 4.4.4. Monotonie-Kriterium

Aus physikalischer Sicht muss die Kamera-Antwortkurve (streng) monoton steigend sein. Diese Eigenschaft wird für  $\mathbf{g}$  im Standard-Ansatz nicht weiter verfolgt. Ein Teil dieser Arbeit ist es deshalb auch, das Verfahren um eine Forderung an die Monotonie von  $\mathbf{g}$  zu erweitern und diese zu implementieren (siehe Abschnitt 5.2).



# 5. Mathematische Ausarbeitung

Zur Lösung des LGS aus Gleichung 4.6 soll in dieser Arbeit ein anderes Verfahren verwendet werden. Hierbei wird das Lösen nach  $\mathbf{g}$  und  $E_i$  in zwei Probleme zerlegt und durch ein alternierendes Lösungsverfahren ersetzt. 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 im Algorithmus 5.1 beschrieben.

Der Vorteil dieses Vorgehens ist, dass neben der Schätzung der Kamera-Antwortkurve auch gleichzeitig die Radiance Map des HDR-Bildes mit berechnet wird. Dadurch spart man sich die anschließende Umrechnung der Bildpunkte mittels der Funktion  $\mathbf{g}$  und ist darüber hinaus auch in der Lage Forderungen an  $\mathbf{E}$  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 aus Gleichung 4.6 dient als Grundlage für den Optimierungsansatz des gesamten Verfahrens. Da dieses Energiefunktional minimiert werden soll, sind partielle Ableitungen nach  $\mathbf{g}(k)$  bzw.  $\ln E_i = F_i$  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 (siehe Gleichung 5.2).

---

### Algorithmus 5.1 Alternierendes Lösen nach $\mathbf{g}(k)$ und $E_i$

---

```
function SOLVEHDR( $Z_{ij}$ ,  $\ln \Delta t_j$ ,  $N$ ,  $P$ )
     $\mathbf{g} \leftarrow initG()$ 
    while  $\mathbf{g}$  changes do
         $\mathbf{F} \leftarrow solveF(\mathbf{F}, \mathbf{g}, Z_{ij}, \ln \Delta t_j, N, P)$                                 //  $F_i = \ln E_i$ 
         $\mathbf{g} \leftarrow solveG(\mathbf{F}, \mathbf{g}, Z_{ij}, \ln \Delta t_j, N, P)$ 
    end while
    return [ $\mathbf{g}$ ,  $\mathbf{F}$ ]
end function
```

---

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

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

$$+ \lambda \underbrace{\sum_{z=Z_{min}+1}^{Z_{max}-1} w^2(z) \cdot \overbrace{[\mathbf{g}(z-1) - 2\mathbf{g}(z) + \mathbf{g}(z+1)]^2}^{\text{Diskretisierung von } g''(k)}}_{\Theta}$$

$$\Omega = \Phi + \lambda \Theta \quad (5.3)$$

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 das 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 (siehe Gleichung 5.4).

$$\frac{\partial \Omega}{\partial \mathbf{g}(k)} = \frac{\partial \Phi}{\partial \mathbf{g}(k)} + \frac{\partial \Theta}{\partial \mathbf{g}(k)} \quad (5.4)$$

$$\frac{\partial \Phi}{\partial \mathbf{g}(k)} = 2 \cdot w^2(k) \cdot \sum_{i=1}^N \sum_{j=1}^P [\mathbf{g}(k) - \ln E_i - \ln \Delta t_j] \cdot \delta_{Z_{ij}=k} \quad \delta_{z=k} = \begin{cases} 1 & \text{wenn } z = k \\ 0 & \text{sonst} \end{cases} \quad (5.5)$$

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

Da  $\Omega$  minimiert werden soll, gilt  $\Omega' \stackrel{!}{=} 0 \Rightarrow (\Theta' \stackrel{!}{=} 0 \wedge \Phi' \stackrel{!}{=} 0)$ . Daraus entsteht das lineare Gleichungssystem für den Datenterm in Gleichung 5.9. Die Koeffizienten der Matrix können aus den einzelnen partiellen Ableitungen gewonnen werden.

$$\frac{\partial \Phi}{\partial \mathbf{g}(k)} \stackrel{!}{=} 0 = 2w^2(k) [\mathbf{g}(k) \sum_{i=1}^N \sum_{j=1}^P \delta_{Z_{ij}=k} - \sum_{i=1}^N \sum_{j=1}^P (\ln E_i - \ln \Delta t_j) \delta_{Z_{ij}=k}] \quad (5.7)$$

$$\underbrace{w^2(k) \sum_{i=1}^N \sum_{j=1}^P (\delta_{Z_{ij}=k})}_{\text{Matrixeintrag } a_k} \cdot \mathbf{g}(k) = \underbrace{w^2(k) \sum_{i=1}^N \sum_{j=1}^P (\ln E_i - \ln \Delta t_j) \delta_{Z_{ij}=k}}_{\text{Vektoreintrag } b_k} \quad (5.8)$$

$$\begin{pmatrix} \ddots & 0 & 0 \\ 0 & a_k & 0 \\ 0 & 0 & \ddots \end{pmatrix} \cdot \begin{pmatrix} \vdots \\ \mathbf{g}(k) \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ b_k \\ \vdots \end{pmatrix} \quad (5.9)$$

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  $\Theta$  integriert, 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 verhält sich die partielle Ableitung anders:

$$\frac{\partial \Theta}{\partial \mathbf{g}(0)} = w^2(1) \cdot \mathbf{g}(0) - 2w^2(1) \cdot \mathbf{g}(1) + w^2(1) \cdot \mathbf{g}(2) = 0 \quad (5.10)$$

$$\begin{aligned} \frac{\partial \Theta}{\partial \mathbf{g}(1)} &= -2w^2(1) \cdot \mathbf{g}(0) \\ &\quad + [4w^2(1) + w^2(2)] \cdot \mathbf{g}(1) \\ &\quad - 2[w^2(1) + w^2(2)] \cdot \mathbf{g}(2) \\ &\quad + w^2(2) \cdot \mathbf{g}(3) \\ &= 0 \end{aligned} \quad (5.11)$$

$$\begin{aligned} \frac{\partial \Theta}{\partial \mathbf{g}(k)} &= w^2(k-1) \cdot \mathbf{g}(k-2) \\ &\quad - 2[w^2(k-1) + 2w^2(k)] \cdot \mathbf{g}(k-1) \\ &\quad + [w^2(k-1) + 4w^2(k) + w^2(k+1)] \cdot \mathbf{g}(k) \\ &\quad - 2[w^2(k) + w^2(k+1)] \cdot \mathbf{g}(k+1) \\ &\quad + w^2(k+1) \cdot \mathbf{g}(k+1) \\ &= 0 \quad \forall k \in [2, 253] \end{aligned} \quad (5.12)$$

$$\begin{aligned} \frac{\partial \Theta}{\partial \mathbf{g}(254)} &= w^2(253) \cdot \mathbf{g}(252) \\ &\quad - 2(w^2(253) + w^2(254)) \cdot \mathbf{g}(253) \\ &\quad + (w^2(253) + 4w^2(254)) \cdot \mathbf{g}(254) \\ &\quad - 2w^2(254) \cdot \mathbf{g}(255) \\ &= 0 \end{aligned} \quad (5.13)$$

$$\frac{\partial \Theta}{\partial \mathbf{g}(255)} = w^2(254) \cdot \mathbf{g}(253) - 2w^2(254) \cdot \mathbf{g}(254) + w^2(254) \cdot \mathbf{g}(255) = 0 \quad (5.14)$$

Aus diesen Gleichungen kann nun das lineare Gleichungssystem (siehe Gleichung 5.15) aufgestellt werden. 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 integriert (siehe Gleichung 5.2).

$$\lambda \underbrace{\begin{pmatrix} d_{0,0} & d_{0,1} & d_{0,2} & 0 & \cdots & & \\ d_{1,-1} & d_{1,0} & d_{1,1} & d_{1,2} & 0 & \cdots & \\ & \ddots & & & & & \\ \cdots & d_{k,-2} & d_{k,-1} & d_{k,0} & d_{k,1} & d_{k,2} & \cdots \\ & & & & \ddots & & \\ & & & d_{254,-2} & d_{254,-1} & d_{254,0} & d_{254,1} \\ & & & d_{255,-2} & d_{255,-1} & d_{255,0} & \end{pmatrix}}_{\text{Matrix } D_4} \cdot \begin{pmatrix} \vdots \\ \mathbf{g}(k) \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ 0 \\ \vdots \end{pmatrix} \quad (5.15)$$

Gemeinsam ergeben die Gleichungssysteme für den Datenterm (siehe Gleichung 5.9) und den Glattheitsterm das endgültige Gleichungssystem für  $\mathbf{g}$ , welches ebenfalls in Matrix-Notation aufgestellt werden kann.

$$\underbrace{[A + \lambda D_4] \cdot \mathbf{g}}_{\text{Matrix } M} = b \quad (5.16)$$

Zu beachten ist, dass  $M$  eine Pentadiagonal-Matrix 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  $E$  dazu, dass die ursprünglich erwähnte Eigenschaft der unendlichen Anzahl von Lösungen (siehe Unterabschnitt 4.1.3) nicht mehr besteht. Dies ist erst während der Implementierung des Ansatzes aufgefallen und liegt daran, dass die beiden Schritte des Verfahrens separat und alternierend ausgeführt werden und immer eine konkrete Näherungslösung der jeweils anderen Unbekannten vorliegen muss. 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.

$$\tilde{\mathbf{g}}(k) = \mathbf{g}(k) - \mathbf{g}(Z_{mid}) \quad \forall k \in [0, 255] \quad (5.17)$$

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

### 5.1.2. Lösen von $E$

Für  $E$  muss in der Fassung des Algorithmus ohne räumlichen Glattheitsterm (siehe Abschnitt 5.3) kein lineares Gleichungssystem gelöst werden. Hier können die Werte direkt

## 5. Mathematische Ausarbeitung

---

berechnet werden, da  $\mathbf{g}$  bekannt ist. Dazu wird das Energiefunktional aus Gleichung 5.2 nach  $\ln E_i = F_i$  partiell abgeleitet und umgeformt.

$$\ln E_i = F_i = \frac{\sum_{j=0}^{P-1} (\mathbf{g}(Z_{ij}) - \ln \Delta t_j) \cdot w(Z_{ij})}{\sum_{j=0}^{P-1} w(Z_{ij})} \quad (5.18)$$

### 5.2. Erweiterung um Monotonie-Eigenschaft

Bereits im Ansatz von Debevec und Malik wird für die Funktion  $f$  angenommen, dass sie monoton und damit invertierbar ist. Für die Funktion  $\mathbf{g}$  wird diese Forderung jedoch nicht weiter aufgenommen. Aus physikalischer und mathematischer Sicht muss  $\mathbf{g}$  jedoch ebenfalls monoton sein, da ansonsten die Annahme  $f$  sei monoton nicht gelten würde. Deshalb wurde eine erste Erweiterung des Algorithmus mit einer Forderung an die Monotonie von  $\mathbf{g}$  realisiert. Dies lässt sich über das Energiefunktional  $\Omega$  als weiteren Bestrafungsterm  $\Gamma$  realisieren.

$$\tilde{\Omega} = \Omega + \mu \underbrace{\sum_{z=1}^{255} w^2(z) [(\phi_{\mathbf{g}'<0}(z) \cdot \mathbf{g}'(z)]^2}_{\text{Monotonie-Forderung } \Gamma} = \Omega + \Gamma \quad (5.19)$$

$$\phi_{\mathbf{g}'<0}(z) = \begin{cases} 1, & \text{falls } \mathbf{g}'(z) < 0 \\ 0 & \text{sonst} \end{cases} \quad (5.20)$$

Auch hier wird wieder der Least-Square-Ansatz (quadratische Bestrafungsfunktion) verwendet, welcher auch schon im Standard-Verfahren zum Einsatz kommt (siehe Gleichung 4.6). Der Operator  $\phi_{\mathbf{g}'<0}(z)$  sorgt dafür, dass nur die Werte von  $\mathbf{g}$  bestraft werden, die nicht monoton steigend sind. Da  $\mathbf{g}'(z)$  bei der Berechnung von  $\mathbf{g}$  jedoch nicht bekannt ist, wird für diese Einschaltfunktion die Instanz  $\mathbf{g}$  aus der vorherigen Iteration verwendet. Durch die Diskretisierung mit  $\mathbf{g}'(z) = \mathbf{g}(z) - \mathbf{g}(z-1)$  erhält man damit:

$$\Gamma = \mu \sum_{z=1}^{255} w^2(z) \cdot \phi_{g'<0}^2(z) \cdot (\mathbf{g}(z) - \mathbf{g}(z-1))^2 \quad (\text{Disketisierung}) \quad (5.21)$$

$$\frac{\partial \Gamma}{\partial \mathbf{g}(0)} = -2\mu w^2(1) \cdot \phi_{g'<0}^2(1) \cdot (\mathbf{g}(1) - \mathbf{g}(0)) \quad (5.22)$$

$$\begin{aligned} \frac{\partial \Gamma}{\partial \mathbf{g}(k)} &= 2\mu w^2(k) \cdot \phi_{g'<0}^2(k) \cdot (\mathbf{g}(k) - \mathbf{g}(k-1)) \\ &\quad - 2\mu w^2(k+1) \cdot \phi_{g'<0}^2(k+1) \cdot (\mathbf{g}(k+1) - \mathbf{g}(k)), \quad \forall k \in [1, 254] \end{aligned} \quad (5.23)$$

$$\frac{\partial \Gamma}{\partial \mathbf{g}(255)} = -2\mu w^2(255) \cdot \phi_{g'<0}^2(255) \cdot (\mathbf{g}(255) - \mathbf{g}(254)) \quad (5.24)$$

Aus dieser Herleitung lässt sich nun wieder eine Matrix mit folgender Struktur erzeugen (hier wurde  $\phi_{g'<0}^2(k) = \phi_{g'}^2(k)$  zur Kürzung verwendet):

$$2\mu \underbrace{\begin{pmatrix} -w^2(1)\phi_{g'}^2(1) & w^2(1)\phi_{g'}^2(1) & 0 & \dots \\ \ddots & \ddots & \ddots & \ddots \\ \dots & -w^2(k)\phi_{g'}^2(k) & \begin{matrix} w^2(k)\phi_{g'}^2(k) \\ +w^2(k+1)\phi_{g'}^2(k+1) \end{matrix} & -w^2(k+1)\phi_{g'}^2(k+1) & \dots \\ \ddots & \ddots & 0 & w^2(255)\phi_{g'}^2(255) & -w^2(255)\phi_{g'}^2(255) \end{pmatrix}}_{\text{Matrix } C} \quad (5.25)$$

Diese Berechnung kann auch über Matritzenmultiplikation erreicht werden.  $D$  ist dabei eine Matrix, welche die erste Ableitung approximiert.  $V$  ist eine Diagonal-Matrix mit  $v_{i,i} = \phi_{g'<0}(i)$ .  $W$  ist die Diagonal-Matrix der Gewichte mit  $w_{i,i} = w(i)$ . Die damit entstehende Gleichung 5.26 kann dann partiell zur Gleichung 5.27 abgeleitet werden.

$$\Gamma = \mu \cdot (WVD\mathbf{g})^2 = \mu(\mathbf{g}^T \cdot D^T V^T W^T WVD \cdot \mathbf{g}) \quad (5.26)$$

$$\frac{\partial \Gamma}{\partial \mathbf{g}} = 2\mu \underbrace{D^T V^T W^T WVD}_{\text{Matrix } C} \cdot \mathbf{g} \stackrel{!}{=} 0 \quad (5.27)$$

Der Parameter  $\mu$  ist ähnlich wie  $\lambda$  ein Gewichtungsfaktor für die Monotonie-Bedingung. Die Gleichung 5.27 kann damit einfach zum Gleichungssystem aus 5.16 hinzugenommen. Daraus entsteht folgendes zu lösende Gleichungssystem:

$$[M + \mu C] \cdot \mathbf{g} = b \quad (5.28)$$

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

### 5.3. Räumlicher Glattheitsterm

Die Erweiterung um den räumlichen Glattheitsterm (also eine Forderung an  $\mathbf{E}$  sich im zweidimensionalen Bild möglichst glatt zu verhalten) ist eine sinnvolle Erweiterung, um die Berechnung der  $\ln E_i$  noch weiter zu optimieren und das lokale Umfeld um einen Bildpunkt mit in die Berechnung einzubeziehen. Dazu fordern wir eine räumliche Glattheit, die (analog zum Glattheitsterm von  $\mathbf{g}$ ) mittels der ersten Ableitung ausgedrückt werden kann. Da wir uns nun jedoch im zweidimensionalen Bildbereich befinden, müssen die Ableitungen in  $x$ - und  $y$ -Richtung betrachtet werden. Der Vektor  $\mathbf{E}$  ist eine eindimensionale Darstellung des zweidimensionalen Bildbereiches ( $n \times m$ ), wobei gilt:  $i = x + y * n$  ( $x \in [0, n]$ ,  $y \in [0, m]$ ). Hierbei müssen die Ränder entsprechend behandelt werden.

$$\tilde{\Omega} = \Phi + \Theta + \alpha \underbrace{\sum_{i \in A} (\overbrace{\ln E_i - \ln E_{i-1}}^{\text{Abltg. nach } x})^2 + \alpha \sum_{i=n}^{N-1} (\overbrace{\ln E_i - \ln E_{i-n}}^{\text{Abltg. nach } y})^2}_{\text{Glattheitsterm } \Psi} \quad (5.29)$$

$$A = \{i \in [0, N-1] \} \setminus \{i \cdot k | k \in \mathbb{N}_+^0\} \quad (5.30)$$

Dies muss nun wieder nach  $\ln E_i$  partiell abgeleitet werden um das Minimum des Energiefunktionalen (siehe Gleichung 5.29) zu bestimmen. Auch hier wurde  $\ln E_i$  durch  $F_i$  ersetzt. Zunächst werden die Randbedingungen vernachlässigt.

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

$$= 2\alpha[4F_i - F_{i-n} - F_{i-1} - F_{i+1} - F_{i+n}] \quad (5.32)$$

Der Einfluss der benachbarten Bildpunkte (siehe Abbildung 5.1) erinnert an einen Hochpassfilter (eng. highpass filter), der in der Bildverarbeitung dazu verwendet wird, verschwommene (engl. blurry) Bilder zu verbessern.

		-1
-1	4	-1
	-1	

**Abbildung 5.1.:** Einfluss der umliegenden Bildpunkte  $F_i$  beim aktiviertem räumlichen Glattheitsterm. Hier in 2D dargestellt

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

$$\frac{\partial \Theta}{\partial F} = 0 \quad (5.33)$$

$$\Phi = \sum_{i=0}^{N-1} \sum_{j=0}^{P-1} w^2(Z_{ij}) [\mathbf{g}(Z_{ij}) - F_i - \ln \Delta t_j]^2 \quad (5.34)$$

$$\frac{\partial \Phi}{\partial F_k} = 2 \sum_{j=0}^{P-1} w^2(Z_{kj}) [\mathbf{g}(Z_{kj}) - F_k - \ln \Delta t_j] \quad (5.35)$$

$$= 2 \sum_{j=0}^{P-1} w^2(Z_{kj}) [\mathbf{g}(Z_{kj}) - \ln \Delta t_j] - 2 \sum_{j=0}^{P-1} w^2(Z_{kj}) F_k \stackrel{!}{=} 0 \quad (5.36)$$

$$\Rightarrow \underbrace{2 \sum_{j=0}^{P-1} w^2(Z_{kj}) [\mathbf{g}(Z_{kj}) - \ln \Delta t_j]}_{\text{Vektoreintrag } b_k} = 2 \underbrace{\sum_{j=0}^{P-1} w^2(Z_{kj})}_{\text{Matrixeintrag } H_{k,k}} F_k \quad (5.37)$$

$$\Rightarrow 2\mathbf{b} = 2H \cdot \mathbf{F} \quad (5.38)$$

Aus der obigen Gleichung 5.31 (in der die Ränder noch nicht beachtet wurden) kann nun die Matrix für die Einbindung der räumlichen Glattheitsforderung erstellt werden (Struktur siehe Abbildung 5.2).

Die entstehende Matrix  $R$  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).

Aus den Gleichungen 5.31 und 5.36 können nun die einzelnen Gleichungen für  $\mathbf{F}_k$  erstellt werden:

$$0 = 2 \sum_{j=0}^{P-1} w^2(Z_{kj}) [\mathbf{g}(Z_{kj}) - \ln \Delta t_j] - 2 \sum_{j=0}^{P-1} w^2(Z_{kj}) F_k + 2\alpha [4F_k - F_{k-n} - F_{k-1} - F_{k+1} - F_{k+n}] \quad (5.39)$$

Aus der Gleichung 5.38 und der Struktur der Matrix  $R$  (siehe Abbildung 5.2) lässt sich damit das nachfolgende Gleichungssystem für die Berechnung von  $\mathbf{F}$  aufstellen.

$$2H \cdot \mathbf{F} + 2\alpha R \cdot \mathbf{F} = 2\mathbf{b} \quad (5.40)$$

$$(H + \alpha R) \cdot \mathbf{F} = \mathbf{b} \quad (5.41)$$

Das Gleichungssystem aus 5.41 ist symmetrisch, quadratisch und positiv semi definit. Aufgrund dieser Eigenschaften kann beim Lösen des LGS das schnell konvergierende SOR-Verfahren (siehe Unterabschnitt 5.5.2) verwendet werden.

$$\left( \begin{array}{cccc|ccc|c} 2 & -1 & & -1 & & & & \\ -1 & 3 & -1 & & -1 & & & \\ & -1 & 3 & -1 & & -1 & & \\ & & -1 & 2 & & & -1 & \\ \hline -1 & & & 3 & -1 & & -1 & \\ & -1 & & -1 & 4 & -1 & & \\ & & -1 & & 4 & -1 & -1 & \\ & & & -1 & & 3 & & \\ \hline & & & -1 & & 3 & -1 & -1 \\ & & & & -1 & 4 & -1 & -1 \\ & & & & & -1 & 4 & -1 \\ & & & & & & -1 & 3 \\ \hline & & & & -1 & & 2 & \\ & & & & & -1 & -1 & \\ & & & & & & -1 & \\ & & & & & & & -1 \end{array} \right)$$

**Abbildung 5.2.:** Schematischer Aufbau der Matrix  $R$  für die Berechnung von  $\ln E_i$  mit räumlicher Glattheit am Beispiel eines  $4 \times 4$  Bildes.

## 5.4. Erweiterung um Robustheit

Wie in 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 (künstlich erzeugt oder z.B. durch Unschärfe bei der Aufnahme der Bilder). Bei anderen Messungenauigkeiten (wie z.B. Salt & Pepper Rauschen) ist ein subquadratischer Bestrafungsterm aus Sicht der Robustheit des Verfahrens jedoch besser geeignet, da hier die starken Ausreißer weniger stark bestrafend wirken. Um die Erweiterung um die Robustheit einzuführen, werden die quadratischen Bestrafungsterme an den gewünschten Stellen durch die subquadratischen ersetzt.

### 5.4.1. Subquadratische Bestrafungsfunktion im Monotonie- oder Glattheits-Term von $g$

An den Termen für die Glattheit von  $g$  und die Monotonie-Forderung an  $g$  (siehe Abschnitt 5.2) ergeben die subquadratischen Bestrafungsterme keinen besonderen Sinn, da diese hier zu stückweise linearen Kurven bzw. stückweise monotonen Funktionen führen würden. Aus diesem Grund wurden diese Terme nicht erweitert.

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

Der Datenterm von  $\mathbf{g}$  berücksichtigt bisher keine Ausreißer. Sind also starke Ausreißer in den Bildern der Belichtungsreihe zu finden (wie z.B. Salt & Pepper 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 das Energiefunktional aus Gleichung 4.6 erweitern. Dieses wird dann wieder partiell nach  $\mathbf{g}(k)$  und  $\ln E_i$  abgeleitet, um den Optimierungsansatz zu lösen.

$$\tilde{\Omega} = \underbrace{\sum_{i=1}^N \sum_{j=1}^P w^2(Z_{ij}) \cdot \varphi'([\mathbf{g}(Z_{ij}) - \ln E_i - \ln \Delta t_j]^2)}_{\text{Datenterm mit Robustheit } \Phi} + \lambda \underbrace{\sum_{z=Z_{\min}+1}^{Z_{\max}-1} [w(Z_{ij}) \cdot \mathbf{g}''(z)]^2}_{\text{Glattheitsterm für } g} \quad (5.42)$$

$$\frac{\partial \tilde{\Phi}}{\partial \mathbf{g}(k)} \stackrel{!}{=} 0 = 2w^2(k) \sum_{i=1}^N \sum_{j=1}^P \delta_{Z_{ij}=k} \underbrace{\varphi'([\mathbf{g}(k) - \ln E_i - \ln \Delta t_j]^2)}_{\text{wird festgehalten}} (\mathbf{g}(k) - \ln E_i - \ln \Delta t_j) \quad (5.43)$$

$$\underbrace{2w^2(k) \sum_{i=1}^N \sum_{j=1}^P \delta_{Z_{ij}=k} \varphi'([\mathbf{g}(k) - \ln E_i - \ln \Delta t_j]^2) \mathbf{g}(k)}_{\text{Matrixeintrag } \tilde{a}_k} \\ = \underbrace{2w^2(k) \sum_{i=1}^N \sum_{j=1}^P (\ln E_i + \ln \Delta t_j) \varphi'([\mathbf{g}(k) - \ln E_i - \ln \Delta t_j]^2) \delta_{Z_{ij}=k}}_{\text{Vektoreintrag } \tilde{b}_k} \quad (5.44)$$

Der dabei vorkommende Bestrafungsfaktor  $\varphi'([\mathbf{g}(k) - \ln E_i - \ln \Delta t_j]^2)$  wird regelmäßig neu berechnet (aus den alten Werten von  $\mathbf{g}$  und  $\ln E_i$ ) und kann zeitweise festgehalten. Damit fließt dieser nur als Faktor in die Berechnung ein. Die Koeffizienten  $\tilde{a}_k$  und  $\tilde{b}_k$  ersetzen die Matrix- bzw. Vektoreinträge des Gleichungssystems aus Gleichung 5.9. Der Glattheitsterm  $\Theta$  bleibt zusammen mit seinen partiellen Ableitungen identisch. Auch mit dieser Erweiterung hat sich die Struktur des LGS für  $\mathbf{g}$  nicht verändert und kann deswegen mit der LU-Zerlegung (siehe Unterabschnitt 5.5.1) gelöst werden.

Diese Erweiterung muss nun auch noch bei der Berechnung von  $\ln E_i$  berücksichtigt werden. Auch hierzu wird das Energiefunktional  $\tilde{\Omega}$  (siehe Gleichung 5.42) wieder partiell nach  $\ln E_i$  abgeleitet. Aus der Gleichung 5.18 entsteht dann bei aktiviertem subquadratischem Bestrafungsterm die neue Berechnung von  $\ln E_i$ :

$$\ln E_i = F_i = \frac{\sum_{j=0}^{P-1} w^2(Z_{ij})(\mathbf{g}(Z_{ij}) - \ln \Delta t_j) \varphi'([\mathbf{g}(Z_{ij}) - \ln E_i - \ln \Delta t_j]^2)}{\sum_{j=0}^{P-1} w^2(Z_{ij}) \varphi'([\mathbf{g}(Z_{ij}) - \ln E_i - \ln \Delta t_j]^2)} \quad (5.45)$$

## 5. Mathematische Ausarbeitung

---

**Algorithmus 5.2** Erweitertes alternierendes Lösen nach  $g(k)$  und  $\ln E_i$  mit Haupt- und Inneniterationen

---

```

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

```

---

Es kann eine schnellere Konvergenz erzielt werden, wenn die einzelnen Berechnungen von  $g$  bzw.  $\ln E_i$  noch häufiger iteriert werden. Um diesen Prozess zu beschleunigen wurden neben den bereits bestehenden Hauptiterationen (siehe Algorithmus 5.1) eine weitere Ebene der Iterationen eingeführt. Auf dieser Ebene wird nur das Lösen nach  $g$  bzw.  $\ln E_i$  wiederholt (siehe Algorithmus 5.2). Dieses Verfahren macht nur Sinn, falls das Monotonie-Kriterium (siehe Abschnitt 5.2) oder die Robustheit mittels subquadratischen Bestrafungstermen aktiviert ist, da hier auf die vorherigen Werte der entsprechenden Unbekannten eingegangen wird.

Das Abbruchkriterium der inneren Schleife könnte ebenfalls wie beim Successive Over-Relaxation (dt. Überrelaxationsverfahren) (SOR) Verfahren (siehe Unterabschnitt 5.5.2) über das relative Residuum erfolgen. In der verwendeten Implementierung wurde jedoch aus Gründen der Einfachheit eine maximale Anzahl an inneren Iterationen festgelegt.

### 5.4.3. Subquadratische Bestrafungsfunktion im räumlichen Glattheitsterm von $E$

Die subquadratischen Bestrafungsfunktionen machen auch bei der Betrachtung des räumlichen Glattheitsterms für  $E$  Sinn. Durch die weniger starke Gewichtung von starken Ausreißern (wie sie z.B. typischer Weise an Kanten in Bildern vorkommen) können Strukturen im Bild besser erhalten werden.

Als Grundlage gilt hier der Glattheitsterm  $\Psi$  aus der Gleichung 5.29. Dieser wird nun um die subquadratische Bestrafungsfunktion  $\varphi(s^2) = \sqrt{s^2 + \epsilon^2}$  erweitert.

$$\Psi = \alpha \sum_{i \in A} \varphi((\ln E_i - \ln E_{i-1})^2) + \alpha \sum_{i=n}^{N-1} \varphi((\ln E_i - \ln E_{i-n})^2) \quad (5.46)$$

$$A = \{i \in [0, N-1] \} \setminus \{i \cdot k | k \in \mathbb{N}\} \quad (5.47)$$

$-\varphi'((F_i - F_{i-1})^2)$	$\begin{aligned} & -\varphi'((F_i - F_{i-n})^2) \\ & \varphi'((F_i - F_{i-1})^2) + \varphi'((F_{i+1} - F_i)^2) \\ & + \varphi'((F_i - F_{i-n})^2) + \varphi'((F_{i+n} - F_i)^2) \\ & - \varphi'((F_{i+n} - F_i)^2) \end{aligned}$	$\varphi'((F_{i+1} - F_i)^2)$
--------------------------------	---	-------------------------------

**Abbildung 5.3.:** Einfluss der umliegenden Bildpunkte  $F_i$  bei aktiviertem räumlichen Glättungsterm mit der Erweiterung durch einen subquadratischen Bestrafungsterm

Dies muss nun wieder nach  $\ln E_i$  partiell abgeleitet werden. Dabei wurde auch hier  $\ln E_i$  durch  $F_i$  ersetzt. Zunächst werden die Randbedingungen ebenfalls vernachlässigt.

$$\begin{aligned} \frac{\partial \tilde{\Psi}}{\partial F_i} = & 2\alpha [ \\ & + \varphi'((F_i - F_{i-1})^2) \cdot (F_i - F_{i-1}) \\ & - \varphi'((F_{i+1} - F_i)^2) \cdot (F_{i+1} - F_i) \\ & + \varphi'((F_i - F_{i-n})^2) \cdot (F_i - F_{i-n}) \\ & - \varphi'((F_{i+n} - F_i)^2) \cdot (F_{i+n} - F_i) \\ ] & \quad (5.48) \end{aligned}$$

$$\begin{aligned} = & 2\alpha [ \\ & - \varphi'((F_i - F_{i-1})^2) \cdot F_{i-1} \\ & - \varphi'((F_{i+1} - F_i)^2) \cdot F_{i+1} \\ & - \varphi'((F_i - F_{i-n})^2) \cdot F_{i-n} \\ & - \varphi'((F_{i+n} - F_i)^2) \cdot F_{i+n} \\ & + \{ \varphi'((F_i - F_{i-1})^2) + \varphi'((F_{i+1} - F_i)^2) + \varphi'((F_i - F_{i-n})^2) + \varphi'((F_{i+n} - F_i)^2) \} \cdot F_i \\ ] \stackrel{!}{=} & 0 \quad (5.49) \end{aligned}$$

Daraus lässt sich wieder ein Stencil (immer noch ohne Berücksichtigung der Ränder) für den Einfluss der umliegenden Bildpunkte bei der Berechnung von  $\ln E_i$  aufstellen (siehe Abbildung 5.3). Die Struktur der Matrix  $\tilde{R}$  (siehe Abbildung 5.2) ist hier wieder ähnlich. An den Rändern fallen entsprechend die Stencil-Einträge an den Seiten weg und treten damit dann auch nicht im zentralen Pixel auf (dieses enthält die positive Summe der Koeffizienten der Umgebungspixel). Auch in dieser Erweiterung wird  $\varphi'(s^2)$  vorab berechnet und dann regelmäßig aktualisiert.

Das daraus entstehende Gleichungssystem in Matrix-Schreibweise ähnelt dem aus Gleichung 5.41.

$$2H \cdot \mathbf{F} + 2\alpha \tilde{R} \cdot \mathbf{F} = \mathbf{2b} \quad (5.50)$$

$$(H + \alpha \tilde{R}) \cdot \mathbf{F} = \mathbf{b} \quad (5.51)$$

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

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

$$\tilde{\Omega} = \underbrace{\sum_{i=1}^N \sum_{j=1}^P w^2(Z_{ij}) \cdot \varphi([\mathbf{g}(Z_{ij}) - \ln E_i - \ln \Delta t_j]^2)}_{\text{Datenterm mit Robustheit } \Phi} + \lambda \underbrace{\sum_{z=Z_{\min}+1}^{Z_{\max}-1} [w(Z_{ij}) \cdot \mathbf{g}''(z)]^2}_{\text{Glattheitsterm für } g} + \alpha \underbrace{\sum_{i \in A} \varphi((\ln E_i - \ln E_{i-1})^2) + \alpha \sum_{i=n}^{N-1} \varphi((\ln E_i - \ln E_{i-n})^2)}_{\text{räumlicher Glattheitsterm mit Robustheit } \Psi} \quad (5.52)$$

(5.53)

Die Gleichung 5.45 kann ebenfalls so umgeformt werden, dass ein LGS entsteht.

$$2 \underbrace{\sum_{j=0}^{P-1} w^2(Z_{ij}) \varphi'([\mathbf{g}(Z_{ij}) - \ln E_i - \ln \Delta t_j]^2) \ln E_i}_{\text{Matrixeintrag } \tilde{h}_i} = 2 \underbrace{\sum_{j=0}^{P-1} w^2(Z_{ij}) (\mathbf{g}(Z_{ij}) - \ln \Delta t_j) \varphi'([\mathbf{g}(Z_{ij}) - \ln E_i - \ln \Delta t_j]^2)}_{\text{Vektoreintrag } \tilde{b}_i} \quad (5.54)$$

$$2 \underbrace{\begin{pmatrix} \ddots & & \\ & \tilde{h}_i & \\ & & \ddots \end{pmatrix}}_{\text{Matrix } \tilde{H}} \cdot \underbrace{\begin{pmatrix} \vdots \\ \ln E_i \\ \vdots \end{pmatrix}}_{\text{Vektor } \mathbf{F}} = 2 \underbrace{\begin{pmatrix} \vdots \\ \tilde{b}_i \\ \vdots \end{pmatrix}}_{\text{Vektor } \tilde{\mathbf{b}}} \quad (5.55)$$

$$2\tilde{H} \cdot \mathbf{F} = 2\tilde{\mathbf{b}} \quad (5.56)$$

Dieses Gleichungssystem wird nun mit der partiellen Ableitung von  $\tilde{\Psi}$  kombiniert (siehe Gleichung 5.50), was zum nachfolgenden Gleichungssystem führt.

$$2\tilde{H} \cdot \mathbf{F} + 2\alpha\tilde{R} \cdot \mathbf{F} = 2\tilde{\mathbf{b}} \quad (5.57)$$

$$(\tilde{H} + \alpha\tilde{R}) \cdot \mathbf{F} = \tilde{\mathbf{b}} \quad (5.58)$$

Strukturell entspricht dieses LGS dem aus Abschnitt 5.3, da die Matrix  $\tilde{H}$  eine Diagonalmatrix mit positiven Einträgen ist. Damit bleibt das LGS positiv semidefinit und quadratisch. Auch hier kommt das SOR Verfahren zum Einsatz.

## 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 5.16 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, was zur Struktur der Matrizen in Gleichung 5.60 führt.

$$A = L \cdot U \quad (5.59)$$

$$\left( A_{ij} \right) = \begin{pmatrix} 1 & 0 & & & \\ l_1 & 1 & 0 & & \\ k_2 & l_2 & 1 & 0 & \\ 0 & k_3 & l_3 & 1 & 0 \\ \ddots & \ddots & \ddots & \ddots & \\ & 0 & k_n & l_n & 1 \end{pmatrix} \cdot \begin{pmatrix} m_0 & r_0 & p_0 & 0 & & \\ 0 & m_1 & r_1 & p_1 & 0 & \\ \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ & \ddots & \ddots & \ddots & \ddots & p_{n-2} \\ & & \ddots & \ddots & \ddots & r_{n-1} \\ & & & 0 & & m_n \end{pmatrix} \quad (5.60)$$

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 enthaltende Lösen des LGS  $A \cdot \mathbf{x} = \mathbf{b}$  geschieht mittels der Vorwärts-Eliminierung und der Rückwärts-Substitution.

Eine erweiterte Pivotisierung der Spalten ist nicht notwendig, da die Diagonal-Einträge der Matrix bereits die betragsmäßig größten Werte einer Spalte haben. 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) musste außerdem noch ein weiterer Typ Matrix gelöst werden. Das dabei entstehende Gleichungssystem hätte nicht so effizient mit der LU-Zerlegung gelöst werden können, da die dabei entstehende Matrix nicht pentadiagonal ist und außerdem sehr groß ist ( $N \times N$ , wobei  $N$  die Anzahl der Pixel in einem Bild der Belichtungsreihe ist).

Hierfür scheint das SOR Verfahren besser geeignet zu sein. Bei Matrizen, die quadratisch, positiv definit, symmetrischen und dünn besetzt sind, stellt das SOR 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:

$$x_k^{m+1} = (1 - \omega)x_k^m + \frac{\omega}{a_{kk}}(b_k - \sum_{i>k} a_{ki}x_i^m - \sum_{i<k} a_{ki}x_i^{m+1}), \quad k \in [1, n] \quad (5.61)$$

Die Implementierung des Verfahrens verwendet als Abbruchkriterium die maximale Komponente  $r_{max}$  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 diese Arbeit auch mit der eigentlichen Implementierung des erarbeiteten Verfahrens. In diesem Kapitel werden die Anforderungen an die Software sowie das Vorgehen bei der Wahl der Programmiersprache beschrieben. Anschließend werden die Installation und Bedienung kurz erklärt. Es folgt eine Auflistung der externen Bibliotheken die zum Einsatz kamen und eine Beschreibung der Architektur der Software.

## 6.1. Anforderungen

Die Anforderungen an die Realisierung der Arbeit 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 unterscheiden [LL10, S. 369f].

### 6.1.1. Funktionale Anforderungen

**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
- Darstellen der Antwortkurve  $g$  als grafischen Plot
- Anzeigen des HDR-Bildes mit verschiedenen Tone-Mapping-Operatoren
- Ändern 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, Mac OS 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, bestand eine gewisse Anforderung an die Qualität des Quellcodes. Diese beinhaltet die folgenden Aspekte.

- Automatisierte Modul-Tests (Zielwert: 90% Anweisungsüberdeckung für das Paket Maths)
- Objektorientierte Programmierung (OOP)
- Die Module und public Funktionen werden gemäß dem offiziellen Javadoc Style Guide<sup>1</sup> kommentiert
- Erstellung einer Software-Dokumentation (z.B. JavaDoc)

**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
- Eine Einweisung soll nicht erforderlich sein
- Validierung der Eingaben des Benutzers
- Gliederung des Programms in Pakete

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.

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

## 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 drei Sprachen Java, C++ und Free Pascal unterscheiden sich dann nur noch kaum. 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 Punkt native Systemzugriff und damit der Performance 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.:** Vergleich der Programmiersprachen mit den Anforderungen. (✓: *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. fpGUI (<http://wiki.freepascal.org/fpGUI>)

<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 Projekte erlangten Vorkenntnissen in Java und des besseren Abschneidens im Vergleich mit den anderen hier genannten Sprachen, wird die Software in Java realisiert.

## 6.3. Programmvorstellung und Benutzeroberfläche

Für die Realisierung der GUI und der grafische Evaluation der Ergebnisse wurde in der vorliegenden Arbeit eine Swing-Oberfläche (siehe Abbildung 6.1) implementiert, die sowohl

## 6. Realisierung

---

die Eingabe der Parameter und Daten als auch die Ausgabe der verschiedenen Diagramme und Bilder ermöglichen soll. In der nachfolgenden Beschreibung sind die umgesetzten Anforderungen zu erkennen.

Über den Button „Bilder wählen“ können die Eingabebilder gewählt werden. Im dafür entwickelten Datei-Fenster steht dem Benutzer eine Vorschau der Bilder zur Verfügung. 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 im unteren Bereich des Fensters 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 6.4) bei der Eingabe der Parameter stellt sicher, dass der Benutzer nur sinnvolle Daten angibt. Bei Falscheingaben ertönt ein akustisches Warnsignal. Die „Start“ Schaltfläche wird bei fehlenden oder fehlerhaften Angaben deaktiviert. Die Eingabemöglichkeiten werden deaktiviert sobald der Berechnungsprozess gestartet wird. Dies verhindert ungültige Benutzerinteraktionen. Das Anwendungsfenster verfügt zudem über eine Fortschrittsanzeige im unteren Bereich. Detaillierte Informationen über den Fortgang der Berechnung können im LOG-Reiter der Anwendung eingesehen werden.

### 6.3.1. Installation

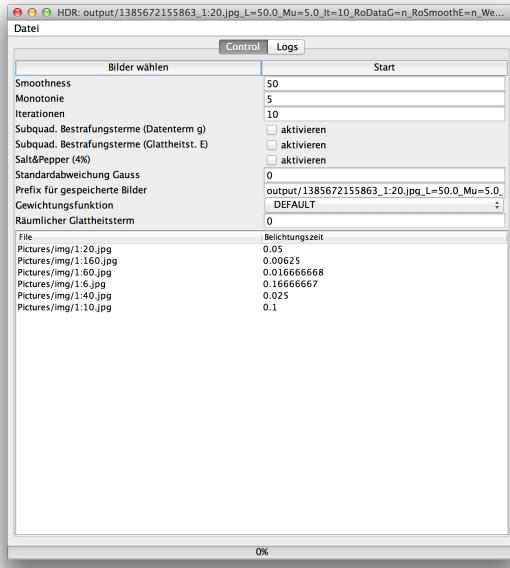
Die Anwendung kann unter <https://github.com/sebastianzillessen/hdr-generator> heruntergeladen werden. Dazu wird ein installiertes JRE der Version 6 oder höher benötigt (siehe <http://www.java.com/de/download/>). Im Unterordner `bin` befindet sich eine ausführbare Version des Programms als `.jar` Datei.

### 6.3.2. Erste Schritte

Nach dem Öffnen der Anwendung erscheint das Hauptfenster (siehe Abbildung 6.1). 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  $\mathbf{g}$
  - Monotonie-Forderung durch die Eingabe des Gewichtes  $\mu$

### 6.3. Programmvorstellung und Benutzeroberfläche



**Abbildung 6.1.:** Die Benutzereingabe für die Erzeugung der HDR-Bilder ist absichtlich schlicht und einfach gehalten.

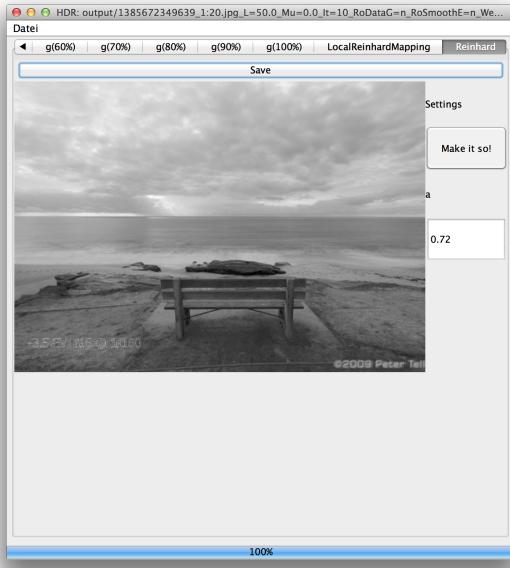
- Anzahl der Haupt- und Inneniterationen (siehe Algorithmus 5.2)
- Aktivierung des Räumlichen Glattheitsterms
- Aktivierung der subquadratischen Bestrafungsterme für **g** und **E**, sowie für den räumlichen Glattheitsterm
- Bildstörungen (Salt & Pepper Rauschen oder additives Gauss-Rauschen) auf den Eingabebildern (zu Testzwecken)
- Änderung des Prefix 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 6.2). Der Export der generierten Bilder und Kurven erfolgt über die Schaltfläche „Speichern“.

## 6. Realisierung

---



**Abbildung 6.2.:** Vorschau des Tone-Mapped Resultats mit der Möglichkeit der Veränderung der Variablen für diesen Tone-Mapper (globaler Tone-Mapping-Operator von Reinhard).

## 6.4. Externe Bibliotheken

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

**NumericTextField<sup>2</sup>** Für die Eingabe der einzelnen Parameter wird eine Implementierung eines numerischen Textfeldes verwendet. Dieses verhindert ungültige Eingaben und erleichtert das Auslesen der numerischen Parameter.

**metadata-extractor<sup>3</sup>** Die Bibliothek `metadata-extractor` wird verwendet um automatisiert aus Bilddateien die Belichtungszeit der Bilder auszulesen. Mit dieser wird auch das von Adobe entwickelte `xmp-core4` mit geliefert. Zusammen ermöglichen es die Bibliotheken die Belichtungszeiten der Bilder auszulesen und darzustellen.

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

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

## 6.5. Architektur

Bei der Architektur der Software wurde eine Model-View-Controller (MVC) Struktur eingehalten. Dieses Muster beschreibt eine klare Trennung zwischen Datendarstellung, -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)*

### 6.5.1. Paketdiagramm

Die Anwendung lässt sich grundsätzlich in fünf verschiedene Komponenten trennen (siehe Abbildung 6.3).

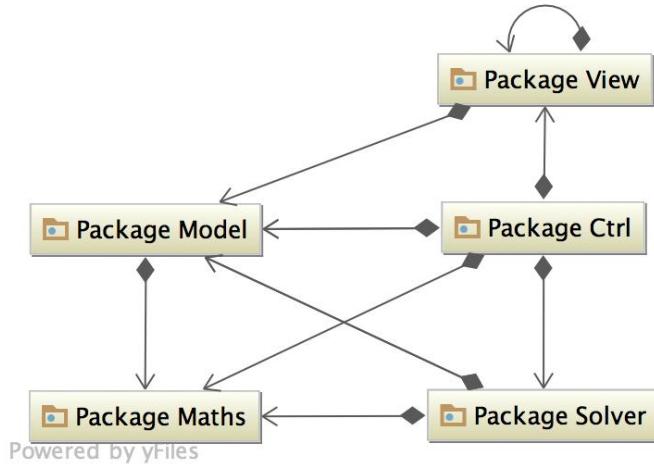
**View:** Das Paket View behandelt die gesamte Darstellung der Anwendung und die grafische Auswertung der berechneten Bilder. Es enthält Tone-Mapping-Operatoren und behandelt die Benutzereingabe.

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

**Ctrl:** Dieses Paket dient zur Steuerung der Anwendung. In diesem werden die Eingaben verarbeitet, an die Algorithmen übergeben und anschließend wieder dargestellt.

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



**Abbildung 6.3.:** Die Architektur der Software auf Komponentenebene. Die Trennung zwischen den einzelnen Bereichen ist hier deutlich erkennbar.

Nachfolgend wird nun auf die einzelnen Komponenten und Klassen näher eingegangen (siehe Abbildung 6.4). 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.

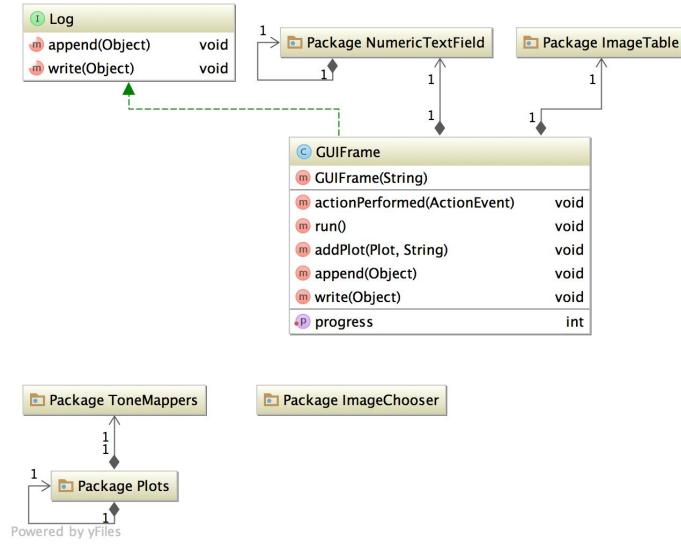
Darin enthalten sind auch die verschiedenen ToneMappers, die zur Darstellung der Radiance Map verwendet werden. Zusammen mit dem Paket Plots wird eine detailliertere Ansicht auf die beteiligten Klassen in Abbildung 6.5 dargestellt.

Einige der mathematischen Aufgaben (wie z.B. die LU-Zerlegung oder das SOR-Verfahren, siehe Abschnitt 5.5) sind im Paket Maths ausgelagert. Die abstrakte Klasse Matrix stellt die Basisfunktionalität zur Verfügung und kennt die beiden Implementierungen BandMatrix und DefaultMatrix. Erstere ist eine spezielle Repräsentation von dünnbesetzten Diagonalmatrizen mit beliebig vielen Bändern. Die hier häufig beschriebenen pentadiagonalen Matrizen 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 DefaultMatrix ist die zweite Implementierung der Matrizen und arbeitet intern mit einem zweidimensionalen Array.

### 6.5.2. Sequenzdiagramm

Das Programm läuft sehr vielschichtig ab. Als grundsätzlicher Informationsfluss dient der Aufbau wie er in Abbildung 6.8 beschrieben. Wichtig dabei ist, dass die grafische Benutzeroberfläche nicht von der Berechnungsduer des Algorithmus blockiert wird, weshalb

## 6.6. Herausforderungen während der Programmierung



**Abbildung 6.4.:** Die View-Komponente im Detail. Das Hauptfenster `GUIFrame` stellt die Daten dar und importiert dazu die verschiedenen Komponenten. Die `Plots` und die `ToneMappers` gehören ebenfalls zu diesem Paket.

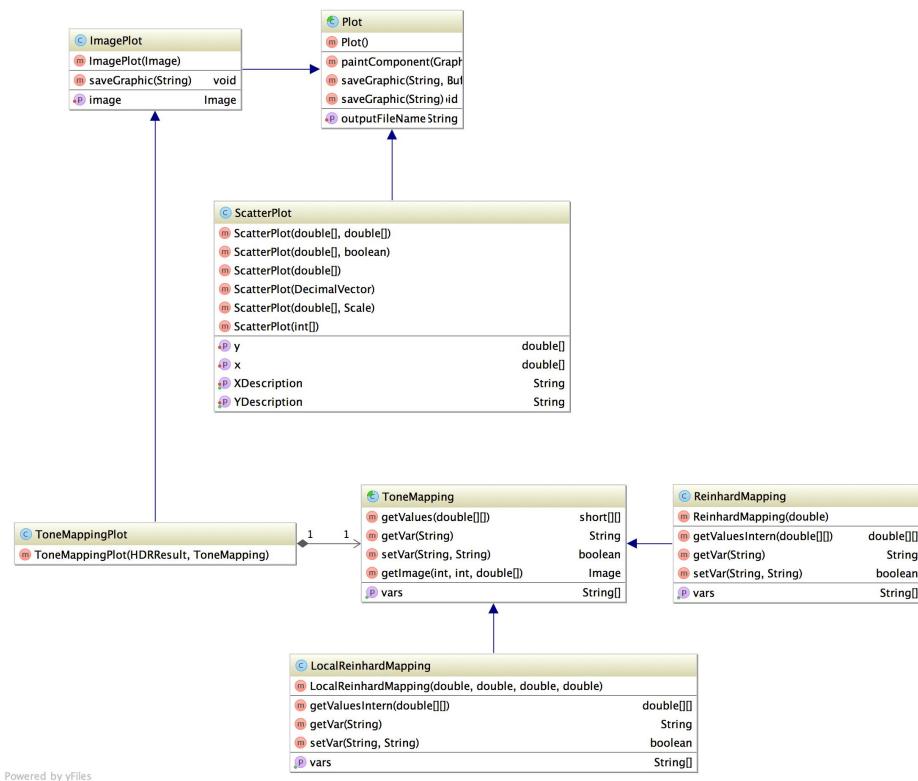
besonders zwischen Controller und `GUIFrame` asynchrone Methodenaufrufe eingesetzt werden.

## 6.6. Herausforderungen während der Programmierung

T.B.D, ggf. rauslassen

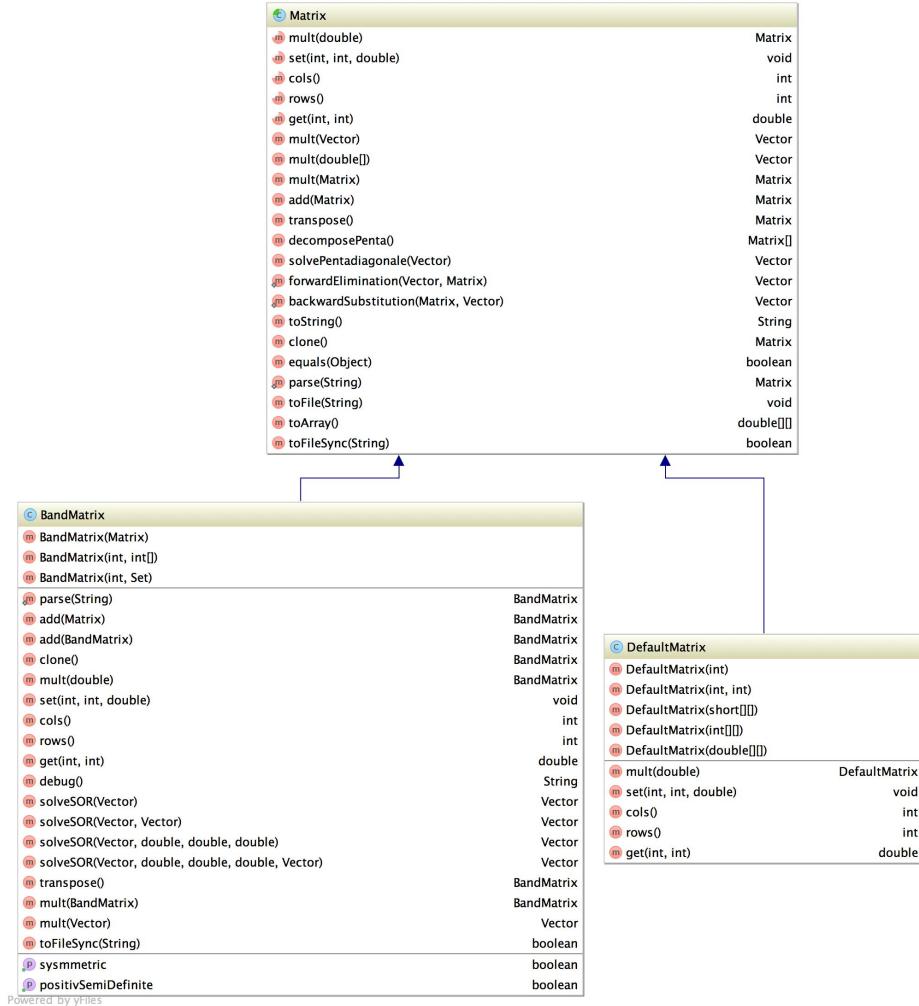
## 6. Realisierung

---



**Abbildung 6.5.:** Die Struktur der verschiedenen grafischen Plots. Die verschiedenen Tone-Mapping-Operatoren wurden mittels Vererbung implementiert.

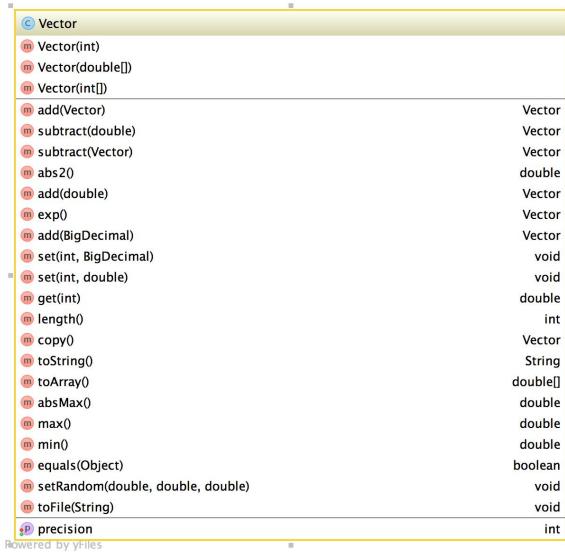
## 6.6. Herausforderungen während der Programmierung



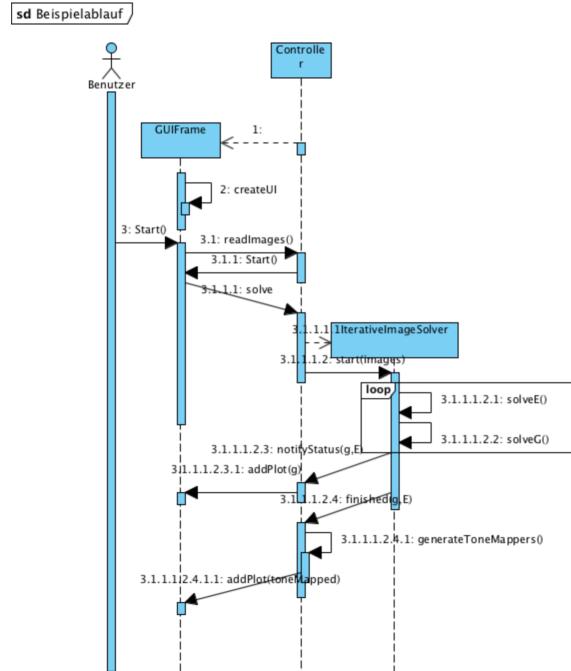
**Abbildung 6.6.:** Die Klasse `Matrix` und ihre Spezialisierungen `BandMatrix` und `DefaultMatrix`. Diese beiden Klassen dienen zusammen mit `Vector` als ein Kernbestandteil der mathematischen Berechnungen dieses Programms.

## 6. Realisierung

---



**Abbildung 6.7.:** Die Klasse `Vector` dient zur Kapselung von Funktionen mit Vektoren und Matrizen.



**Abbildung 6.8.:** Der grundsätzliche Ablauf der Berechnung des HDR-Bildes und der Antwortkurve und der dabei beteiligten Komponenten sowie deren Interaktion.

## 7. Ergebnisse und Resultate

Durch diese Arbeit konnte insbesondere durch die Erweiterung durch den räumlichen Glattheitsterm und die Ergänzung mit subquadratischen Bestrafungstermen eine Verbesserung der Ergebnisse festgestellt werden. Die Implementierung des Standard-Verfahrens vonDebevec und Malik zeigt insbesondere bei der Generierung der Antwortkurve aus allen Bildpunkten Probleme mit der Glattheit der Antwortkurve auf (siehe Abbildung 7.1). Der Vergleich zur herkömmlichen Methode, bei der nur eine begrenzte Anzahl an Bildpunkten verwendet wird, zeigt jedoch wenig Unterschiede in der Antwortkurve.

In den nachfolgenden Beschreibungen werden folgende Symbole mit den angegebenen Standardwerten verwendet, falls nichts anderes dazu angegeben wird:

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

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

$\alpha$  : Gewicht der räumlichen Glattheit für  $E$ , 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, [RSSF02]) verwendet.



**Abbildung 7.1.:** Verfahren der Berechnung der Antwortkurve und dem dazugehörigen Resultat (re.) **links:** Unser Verfahren, **rechts:** Implementierung von Mathias Eitz, siehe Abschnitt 3.1

## 7. Ergebnisse und Resultate

---



**Abbildung 7.2.:** Einfluss der Monotonie-Forderung: Belichtungsserie (Belichtungszeiten:  $1/8000, 1/640, 1/100$ ), Antwortkurve ohne Monotonie-Forderung, Antwortkurve mit Monotonie-Forderung (v.l.n.r.).

Die in dieser Arbeit vorgestellten Erweiterungen werden auf die Belichtungsserie von [Tel10] angewandt um deren Effekte zu zeigen.

### 7.1. Ergebnisse mit Monotonie-Bedingung

Bei den meisten Bildern ist die Antwortkurve  $g$  bereits von sich aus monoton steigend. 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.

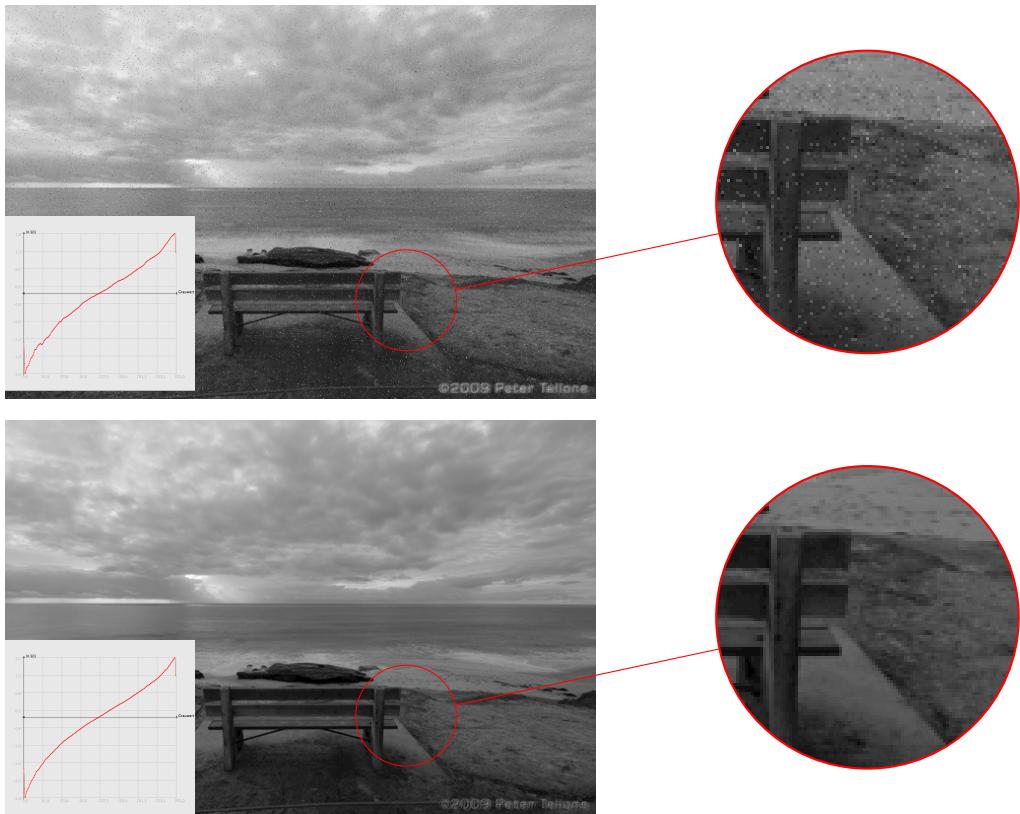
### 7.2. Ergebnisse mit räumlichem Glattheitsterm

Die Erweiterung des räumlichen Glattheitsterms sorgt dafür, dass die benachbarten Pixel bei der Berechnung der Radiance Map berücksichtigt werden. Wie in Abbildung 7.3 zu sehen, hat dies insbesondere bei Messfehlern (hier 4% Salt & Pepper Rauschen) einen enormen Vorteil gegenüber dem herkömmlichen Verfahren. Auch durch Störungen wie das additive Gauss-Rauschen ( $\sigma = 10$ , siehe Abbildung 7.4) sind die Ergebnisse schärfer und das Rauschen auf den Eingangsbildern wird reduziert.

### 7.3. Ergebnisse mit subquadratischen Bestrafungstermen

Die Erweiterung mit subquadratischen Bestrafungsfunktionen liefert unter normalen Umständen kaum eine Verbesserung (siehe Abbildung 7.5), lediglich der Kontrast und die Konturenschärfe wird etwas verbessert. Wird hingegen auch noch Rauschen (hier Salt & Pepper Rauschen, siehe Abbildung 7.6) simuliert, dann ist dies im Ausgabebild quasi nicht mehr

### 7.3. Ergebnisse mit subquadratischen Bestrafungstermen



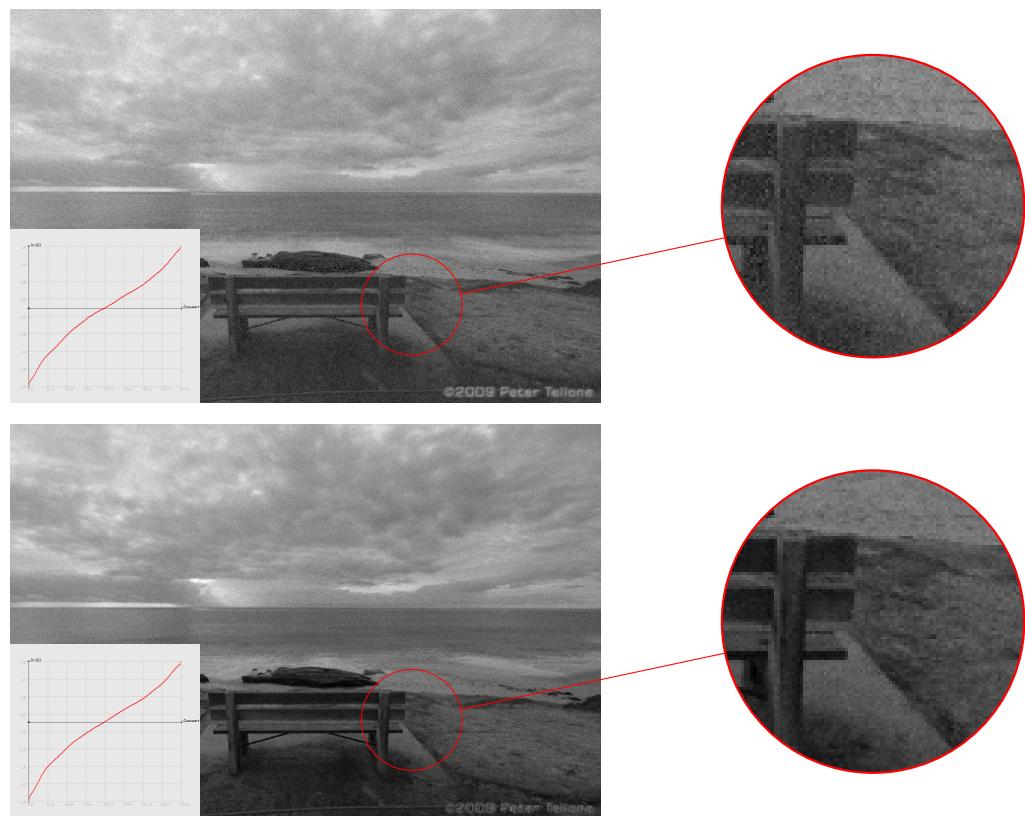
**Abbildung 7.3.:** Einfluss des räumlichen Glattheitsterms auf verrauschte Bilder (4% Salt & Pepper Rauschen): **oben:** Antwortkurve und zugehöriges HDR Bild ohne räumlichen Glattheitsterm, **unten:** Antwortkurve mit räumlichem Glattheitsterm ( $\alpha = 1$ ) und zugehöriges HDR-Bild. Das Salt & Pepper Rauschen ist im unteren Bild quasi nicht mehr zu erkennen.

zu erkennen und es kann eine eindeutige Verbesserung erzielt werden. Dieser Effekt entsteht durch die geringere Gewichtung der Ausreißer (dem Salt & Pepper) bei der Berechnung von  $g$  und  $E$ .

Von der Erweiterung des räumlichen Glattheitsterms um robuste Funktionen wurde eine Verbesserung der Kantenerhaltung erwartet. Diese Erwartung wurde jedoch, wie in Abbildung 7.7 zu erkennen ist, nicht erfüllt. Die Verbesserungen sind trotz hoher Gewichtung so minimal, dass sie kaum zu erkennen sind.

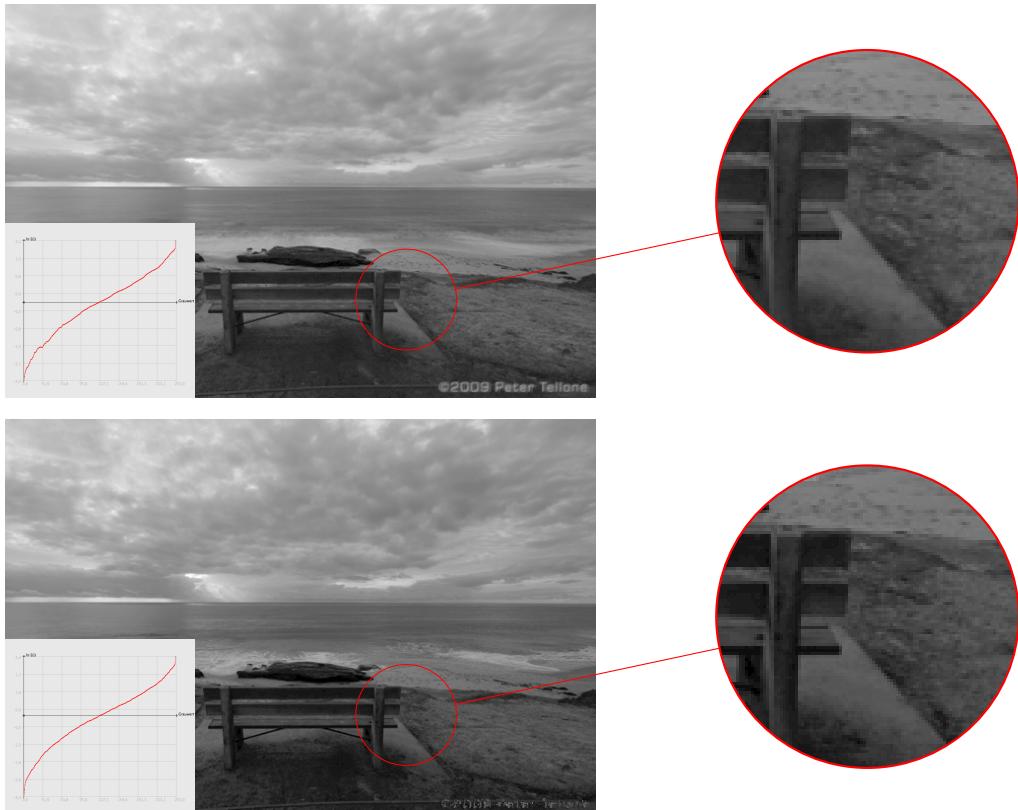
## 7. Ergebnisse und Resultate

---



**Abbildung 7.4.:** Einfluss des räumlichen Glattheitsterms auf verrauschte Bilder (additives Gauss-Rauschen mit  $\sigma = 10$ ): **oben:** Antwortkurve und zugehöriges HDR Bild ohne räumlichen Glattheitsterm, **unten:** Antwortkurve mit räumlichem Glattheitsterm ( $\alpha = 1$ ) und zugehöriges HDR-Bild.

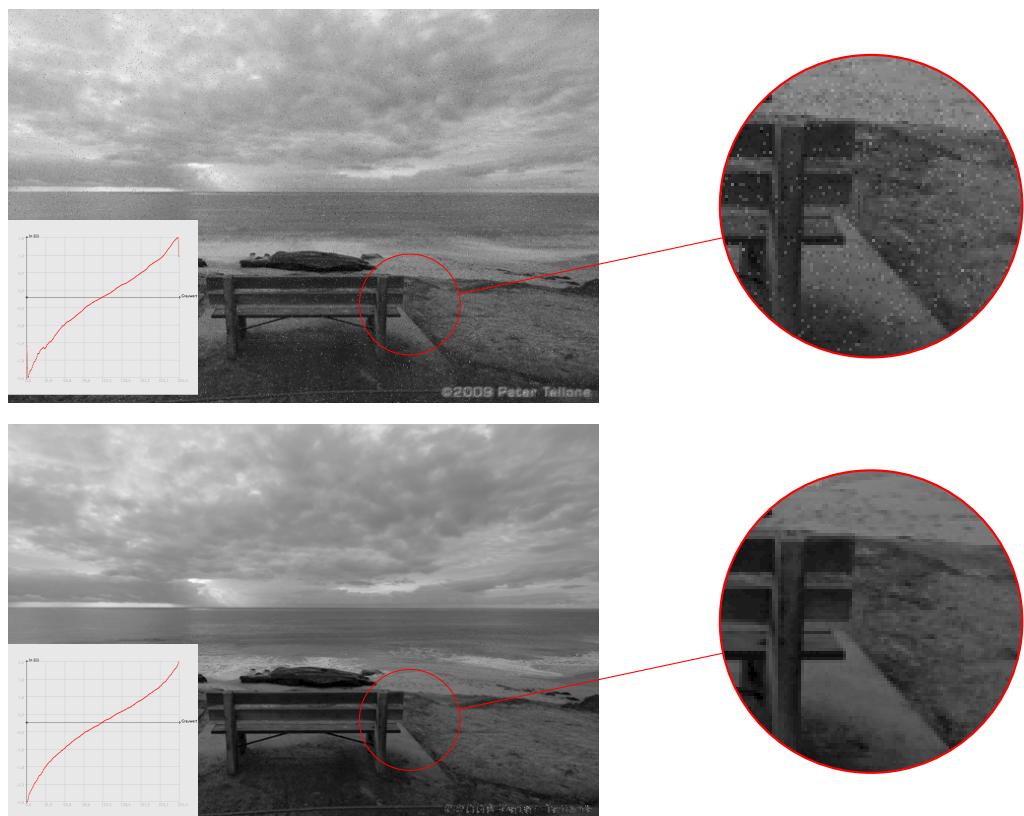
### 7.3. Ergebnisse mit subquadratischen Bestrafungstermen



**Abbildung 7.5.:** Einfluss der subquadratischen Bestrafungsfunktionen: **oben:** Standardverfahren, **unten:** Subquadratischer Bestrafungsterm  $\varphi(s^2) = \sqrt{s^2 + \epsilon^2}$  für **g** und **E**, die Kanten sind schärfer und der Kontrast besser

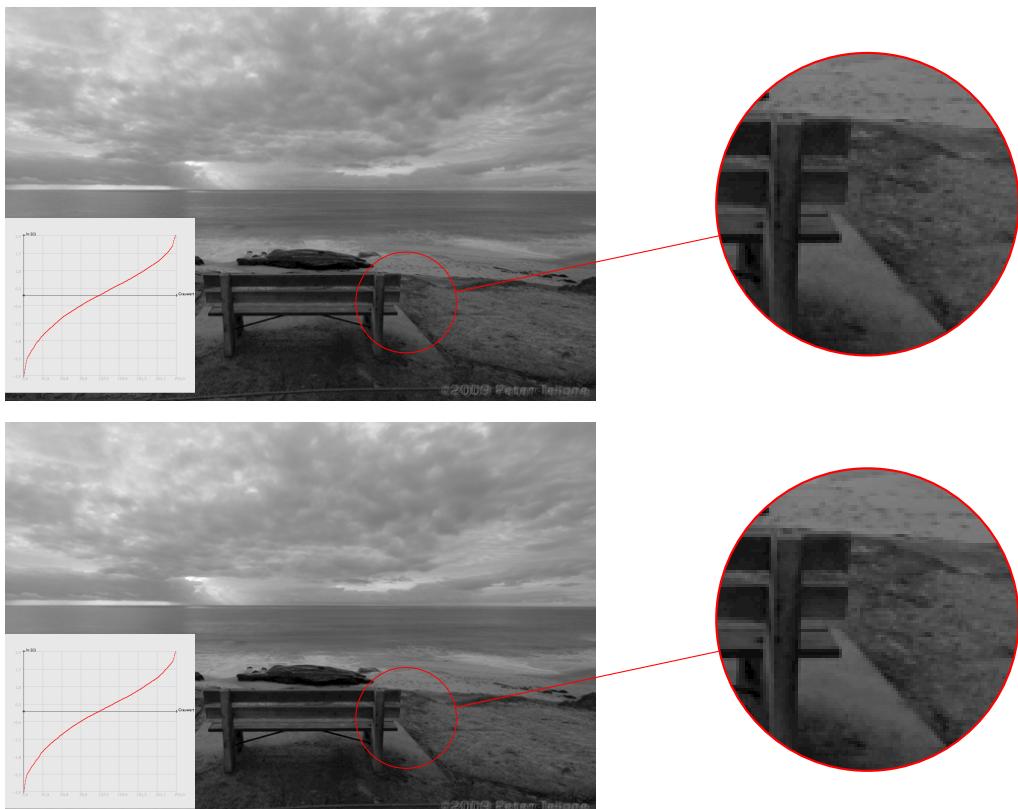
## 7. Ergebnisse und Resultate

---



**Abbildung 7.6.:** Einfluss der 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.

### 7.3. Ergebnisse mit subquadratischen Bestrafungstermen



**Abbildung 7.7.:** Einfluss der subquadratischen Bestrafungsfunktionen auf den räumlichen Glattheitsterm bei Salt & Pepper Rauschen ( $\alpha = 100$ ): **oben:** Ohne subquadratische Bestrafungsfunktionen, **unten:** Mit subquadratischen Bestrafungsfunktionen. Die Kanten sind nur minimal besser erhalten.



## **8. Zusammenfassung und Ausblick**

T.B.D

### **Ausblick**

T.B.D



# A. Anhang

## A.1. Verwendete Algorithmen

### A.1.1. MTB-Algorithmus, vgl. [War03, S.9 f]

Dem MTB Verfahren (vgl. [War03]) 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  $(x, y)$  ausgeben, sodass die Bilder exakt übereinander gelegt werden können (vgl. [RWPDo5, 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 Bildpyramide zum Einsatz, dass dann mit schnellen Bit-Operationen die Verschiebung der Bilder berechnet (vgl. [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)
```

---

## A. Anhang

---

**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);
}
```

---

### A.1.2. 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
     $L \leftarrow \text{GENERATEL}(l, k)$                                 // Generiert die Matrix L und U
     $U \leftarrow \text{GENERATEU}(m, r, p)$                           // nach obigem Schema (siehe Gleichung 5.60)
    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  $\text{size}(b)$  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
     $[L, U] = \text{LUDECOMPOSITION}(A);$ 
     $y \leftarrow \text{FORWARDELIMINATION}(b, L)$                       // forward elimination  $L^*y = b$ 
     $x \leftarrow \text{BACKWARDSUBSTITUTION}(U, y)$                   // backward substitution  $U^*x = y$ 
    return  $x$ 
end function

```

---



# Glossar und Abkürzungsverzeichnis

## **Belichtungszeit**

—. 16

## **CMOS**

Complementary Metal Oxide Semiconductor. 17, 26

## **Dynamikumfang**

Verhältnis von größter und kleinster Leuchtdichte.. 16, 20

## **GUI**

grafische Benutzerschnittstelle (engl. graphical user interface). 49–51

## **HDR**

High Dynamic Range. 7, 11–13, 15–23, 25–28, 33

## **JRE**

Java Runtime Environment. 51

## **LDR**

Low Dynamic Range. 18, 20, 21, 25, 26

## **LGS**

Lineares Gleichungssystem. 29, 30, 33, 34, 37, 41

## **Monotonie**

Test test. 6, 12, 60

## **MTB**

Mean Threshold Bitmap Alignment. 18, 69

## **MVC**

Model-View-Controller. 52

## **OOP**

Objektorientierte Programmierung. 49–51

**Pentadiagonal-Matrix**

ist eine (analog zu einer Tridiagonal-Matrix) eine quadratische Matrix, bei der nur die fünf zentralen Diagonalen besetzt sind.. 37

**Radiance Map**

Test test. 16, 20, 29, 30, 33, 52, 53, 60

**RAW**

Rohdatenformate. 17

**RGB-Farbraum**

test. 15

**Salt & Pepper Rauschen**

Das sog. Salt & Pepper Rauschen beschreibt eine Verkörnung des Bildes, bei 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. . 7, 8, 12, 42, 43, 55, 60, 61, 64, 65

**SOR**

Successive Over-Relaxation (dt. Überrelaxationsverfahren). 44, 46–48

**SVD**

singular value decomposition (dt. Singulärwertzerlegung). 29

**Tone-Mapping**

Test test. 7, 8, 12, 15, 20–23, 26, 52, 55, 58, 59

# Literaturverzeichnis

- [Ado92] Adobe. Tiff specification, Revision 6.0, 1992. URL <http://partners.adobe.com/public/developer/en/tiff/TIFF6.pdf>. (Zitiert auf Seite 19)
- [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 17)
- [Blo12] C. Bloch. *The HDRI Handbook - High Dynamic Range Imaging for Photographers and CG Artists*. O'Reilly Media, Sebastapool, 1 Auflage, 2012. (Zitiert auf den Seiten 18 und 20)
- [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 31)
- [Debo08] 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 2008 Classes*, S. 32:1–32:10. ACM Press, 2008. (Zitiert auf Seite 17)
- [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 5, 7, 11, 12, 13, 22, 27, 28 und 30)
- [FJo04] 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 16)
- [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 26)
- [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), 2007. (Zitiert auf den Seiten 21 und 26)
- [Lar98] G. W. Larson. LogLuv Encoding for Full-gamut, High-dynamic Range Images. Band 3, S. 15–31. A. K. Peters, Ltd., 1998. (Zitiert auf Seite 20)

- [LG03] X. C. Liu, A. E. Gamal. Synthesis of high dynamic range motion blur free image from multiple captures. Band 50, S. 530–539. IEEE Press, 2003. (Zitiert auf Seite 26)
- [LL10] J. Ludewig, H. Lichter. *Software Engineering*. dpunkt.verlag GmbH, Heidelberg, 2 Auflage, 2010. (Zitiert auf den Seiten 49 und 55)
- [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 18)
- [NMoo] S. K. Nayar, T. Mitsunaga. High dynamic range imaging: Spatially varying pixel exposures. S. 472–479. IEEE Press, 2000. (Zitiert auf Seite 25)
- [RSD05] E. Reinhard, I. C. Society, K. Devlin. Dynamic range reduction inspired by photoreceptor physiology. *IEEE Transactions on Visualization and Computer Graphics*, 11:13–24, 2005. (Zitiert auf den Seiten 7 und 23)
- [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 21 und 61)
- [RWPDo5] E. Reinhard, G. Ward, S. Pattanaik, P. Debevec. *High Dynamic Range Imaging: Acquisition, Display, and Image-Based Lighting*. Morgan Kaufmann Publishers Inc., 2005. (Zitiert auf den Seiten 8, 9, 15, 16, 17, 19, 20 und 71)
- [Tel10] P. Tellone. How to Shoot and Post-Process Professional HDR Photos in One Day, 2010. URL <http://photography.tutsplus.com/tutorials/how-to-shoot-and-post-process-professional-hdr-photos-in-one-day-2--photo-410> (Zitiert auf den Seiten 7, 12, 15, 16, 23 und 62)
- [War03] G. Ward. Fast, robust image registration for compositing high dynamic range photographs from handheld exposures. *Journal of Graphics Tools*, 8:17–30, 2003. (Zitiert auf den Seiten 9, 71 und 72)
- [Wes08] T. Westermann. *Mathematik für Ingenieure - Ein anwendungsorientiertes Lehrbuch*. Springer, 6 Auflage, 2008. (Zitiert auf Seite 48)
- [YBMS05] A. Yoshida, V. Blanz, K. Myszkowski, H.-P. Seidel. Perceptual evaluation of tone mapping operators with real-world scenes, 2005. (Zitiert auf den Seiten 21 und 26)
- [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*, 34:1821–1834, 1999. (Zitiert auf Seite 16)
- [ZBW11] H. Zimmer, A. Bruhn, J. Weickert. Freehand HDR Imaging of moving scenes with simultaneous resolution enhancement. Band 30, S. 405–414. 2011. (Zitiert auf Seite 26)

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