

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

1. Einleitung	11
1.1. Motivation	11
1.2. Aufgabenstellung	12
1.3. Gliederung	13
2. Grundlagen der HDR-Bilder	15
2.1. Prinzip	15
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
3. Verwandte Arbeiten und Implementierungen	25
3.1. Bekannte Implementierungen des Ansatzes von Debevec und Malik	25
3.2. Verwandte Arbeiten	25
4. Algorithmus von Debevec und Malik [DM97]	27
4.1. Ansatz	27
4.2. Berechnung der Antwortkurve	29
4.3. Konstruktion der Radiance Map	29
4.4. Mögliche Erweiterungen des Ansatzes	30
5. Mathematische Ausarbeitung	33
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
6. Implementierung	49
6.1. Wahl der Programmiersprache	49
6.2. Externe Bibliotheken	50
6.3. Architektur	51
6.4. Programmvorstellung	57
6.5. Herausforderungen während der Programmierung	59

7. Ergebnisse und Resultate	61
7.1. Ergebnisse mit Erweiterungen	61
8. Zusammenfassung und Ausblick	69
A. Anhang	71
A.1. Artikel [DM97]	71
A.2. Verwendete Algorithmen	82
Glossar und Abkürzungsverzeichnis	85
Literaturverzeichnis	87

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. o.l.: ohne robuste Bestrafungsterme. o.r.: mit robusten Bestrafungstermen und räumlicher Glattheitsforderung. unten: Das HDR-Bild wurde mittels lokalem Reinhard-Tone-Mapper (vgl. ??) erstellt.	12
2.1.	Beispielhaftes High Dynamic Range (HDR)-Bild mit bereits angewandtem Tone-Mapping-Verfahren (siehe [Tel10])	16
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].	17
2.3.	Der Erstell-Assistent von Luminance HDR. Auch hier wird intern der Algorithmus vonDebevec und Malik [DM97] verwendet.	23
2.4.	Ein mögliches Ergebnis des HDR-Bildes. Erstellt mit Luminance HDR und dem Tone-Mapping-Operator Reinhard'05 [RSD05]. Bildserie siehe [Tel10].	24
5.1.	Einfluss der umliegenden Bildpunkte F_i beim aktiviertem räumlichen Glattheitsterm. 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 4×4 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 Architektur der Software auf Komponentenebene. Die Trennung zwischen den einzelnen Bereichen ist hier deutlich erkennbar.	52
6.2.	Die View-Komponente im Detail. Das Hauptfenster <code>GUIFrame</code> stellt die Daten dar und importiert dazu die verschiedenen Komponenten. Die Plots und die <code>ToneMappers</code> gehören ebenfalls zu diesem Paket.	53
6.3.	Die Struktur der verschiedenen grafischen Plots. Die verschiedenen Tone-Mapping-Operatoren wurden mittels Vererbung implementiert.	54
6.4.	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.	55
6.5.	Die Klasse <code>Vector</code> dient zur Kapselung von Funktionen mit Vektoren und Matrizen.	56

6.6. Der grundsätzliche Ablauf der Berechnung des HDR-Bildes und der Antwortkurve und der dabei beteiligten Komponenten sowie deren Interaktion.	57
6.7. Die Benutzereingabe für die Erzeugung der HDR-Bilder ist absichtlich schlicht und einfach gehalten.	58
6.8. Auswahl der Belichtungsserie incl. Vorschau.	59
6.9. Vorschau des Tone-Mapped Resultats mit der Möglichkeit der Veränderung der Variablen für diesen Tone-Mapper (globaler Tone-Mapping-Operator von Reinhard).	60
7.1. Verfahren der Berechnung der Antwortkurve (li.) und dem dazugehörigen Resultat (re.) oben : Unser Verfahren, unten : Implementierung von Mathias Eitz, siehe Abschnitt 3.1 (keine Erweiterungen aktiviert)	62
7.2. oben : Belichtungsserie, Belichtungszeit: 1/8000, 1/640, 1/100 (v.l.), u.l. : Antwortkurve ohne Monotonie-Forderung, u.r. : Antwortkurve mit Monotonie-Forderung	63
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.	64
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.	65
7.5. Einfluss der subquadratischen Bestrafungsfunktionen: oben : Standardverfahren, mitte : Subquadratischer Bestrafungsterm $\varphi(s^2) = \sqrt{s^2 + \epsilon^2}$ für g und E , die Kanten sind schärfer und der Kontrast besser, unten : Zusätzliches Salt & Pepper Rauschen ist quasi nicht mehr zu erkennen.	66
7.6. Einfluss der subquadratischen Bestrafungsfunktionen: oben : Standardverfahren, mitte : Subquadratischer Bestrafungsterm $\varphi(s^2) = \sqrt{s^2 + \epsilon^2}$ für g und E , die Kanten sind schärfer und der Kontrast besser, unten : Zusätzliches Salt & Pepper Rauschen ist quasi nicht mehr zu erkennen.	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 Java, C und C# mit den Anforderungen.	50

Verzeichnis der Listings

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)	84

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 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.

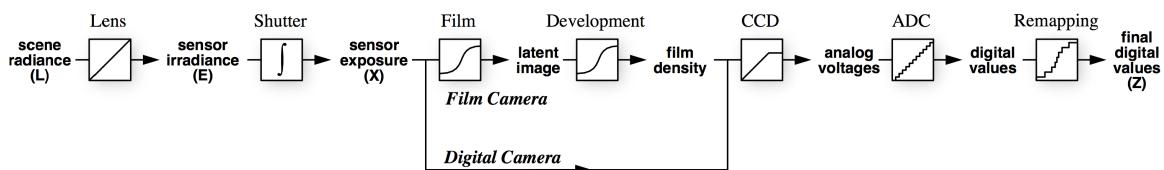


Abbildung 1.1.: Die Bildaufnahme Pipeline veranschaulicht, welche Prozesse durchlaufen werden bis aus einer realen Szene das digitale Bild für die Verwendung erzeugt wird (links nach rechts) [DM97, S.2]

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

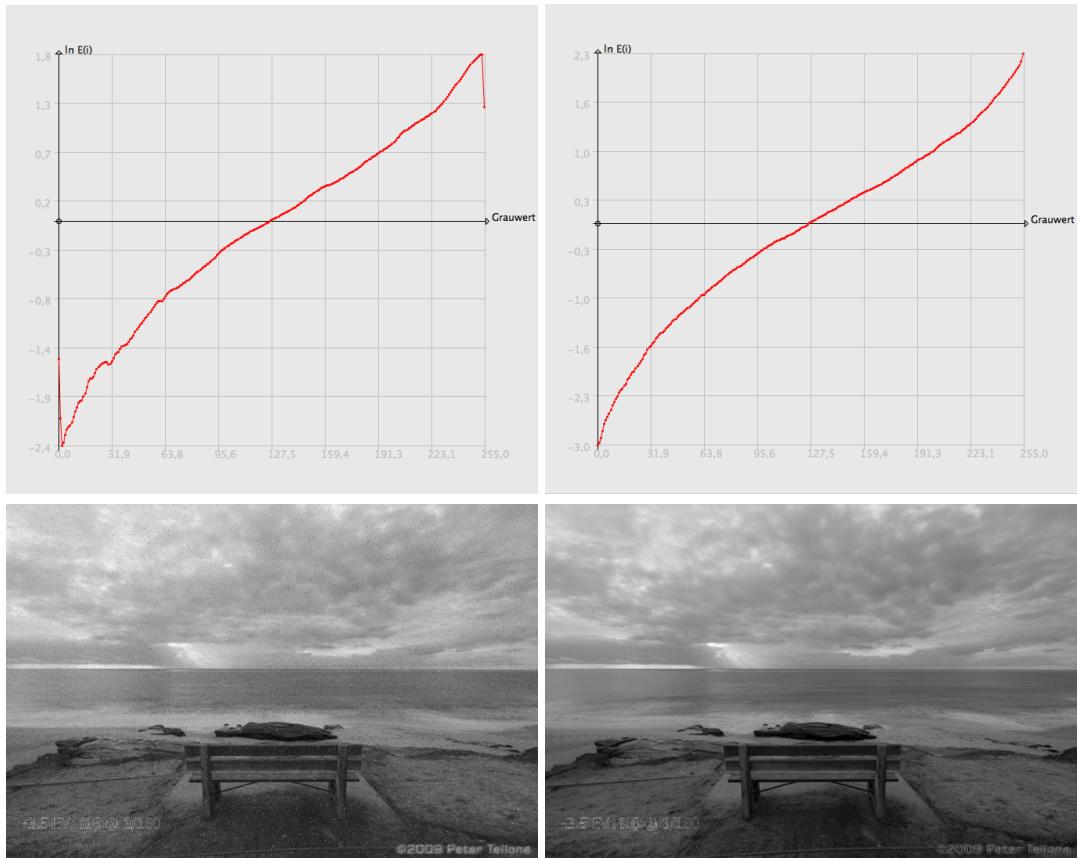


Abbildung 1.2.: Geschätzte Antwortkurven und zugehöriges HDR-Bild für die Bilderserie [Tel10] mit Salt & Pepper Rauschen. **o.l.:** ohne robuste Bestrafungsterme.
o.r.: mit robusten Bestrafungstermen und räumlicher Glattheitsforderung.
unten: Das HDR-Bild wurde mittels lokalem Reinhard-Tone-Mapper (vgl. ??) erstellt.

Neben der Modellierung und Implementierung der einzelnen Erweiterungen soll auch eine geeignete visuelle Evaluation der Ergebnisse erfolgen. Hierzu sollen Tone-Mapping-Verfahren (siehe ??) 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 – Implementierung: Die Implementierung der in Kapitel 4 und Kapitel 5 beschriebenen Modelle wird hier erläutert. Darüber hinaus werden einzelne Programm-Passagen (siehe Abschnitt 6.4) genauer beleuchtet und die Laufzeit der Anwendung (siehe ??) analysiert.

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]).

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 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. [FJo4]) 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



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

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]

Dichte der Bildinformationen (z.B. die großen Helligkeitsunterschiede) aufzunehmen (vgl. [YGFT99]).

Der hier verwendete Begriff „Dynamikumfangs“ 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 Dynamikum-

fang 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 $1/6$, $1/10$, $1/16$, $1/20$, $1/40$, $1/60$ 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 Photoshop¹, Photogenics² und Photomatix³ 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

¹<http://adobe.com/photoshop>

²<http://www.cinepaint.org>

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

2. Grundlagen der HDR-Bilder

Medizin große Fortschritte machen zu können. Solche Sensoren können bereits in der Größe eines Streichholzkopfes einen Dynamikumfang von 179 dB erreichen (vgl. [BGH⁺06]).

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. [Debo08]) können HDR-Bilder eingesetzt werden, um dem Betrachter eine noch „realere“ Szene zu suggestieren.

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

Format	Kodierung	Kompression	Metadaten	Lizenz
HDR	RGBE	Lauflängen-kodierung	Kalibrierung, Farbraum + benutzerdef. Da- ten	Open source soft- ware (<i>Radiance</i>)
	XYZE	Lauflängen-kodierung	+ benutzerdef. Da- ten	
TIFF	IEEE RGB	keine	Kalibrierung, Farbraum + Registrierung + benutzerdef. Da- ten	Public domain li- brary (<i>libtiff</i>)
	LogLuv24	keine	+ benutzerdef. Da- ten	
	LogLuv32	Lauflängen- kodierung		
EXR	Half RGB	Wavelet, ZIP	Kalibrierung, Farbraum + Fensterfunktion + benutzerdef. Da- ten	Open source libra- ry (<i>OpenEXR</i>)

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

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 Kantendetektionsverfahren 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.2.1). Auf eine Implementierung dieses Verfahren wurde verzichtet, da sie nicht relevant für die Zielsetzung sind.

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]. 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 Format⁴ 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

⁴www.openexr.com

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]).

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). α 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 gewählt⁵.

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

⁵TODO: Wahl Operator

2. Grundlagen der HDR-Bilder

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. Photoshop⁶ oder GIMP⁷) 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.

Darüber hinaus gibt es verschiedene Programme, die speziell auf die Erzeugung von HDR-Bildern spezialisiert sind (z.B. Photomatix⁸ oder Luminance HDR⁹). Der Funktionsumfang dieser Programme ist verhältnismäßig klein, führt jedoch auch Laien schnell zum Ziel, da (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 HDR¹⁰ und HDR Darkroom¹¹ 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 Picturenaut¹². Die Anzahl und der Funktionsumfang der implementierten Tone-Mapping-Operatoren ist limitiert, jedoch liefert das Programm recht rasch realitätsgetreue Bilder.

⁶<http://www.adobe.com/de/products/photoshop.html>

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

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

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

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

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

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

2.6. Software zur Erstellung von HDR-Bildern

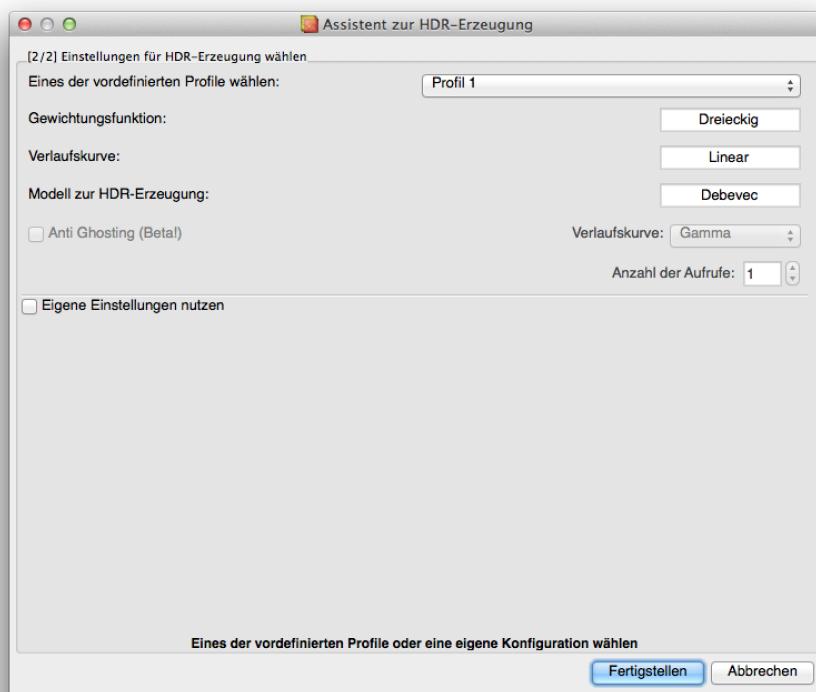


Abbildung 2.3.: Der Erstell-Assistent von Luminance HDR. Auch hier wird intern der Algorithmus von Debevec und Malik [DM97] verwendet.

2. Grundlagen der HDR-Bilder



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

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 Implementierung¹ 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.

¹<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⁺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 $\ln E_i$

Δt_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 Δt 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 α 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 Ω 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 Ω 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 Θ . 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 λ absichtlich nicht in Θ integriert, da dieser bei der Herleitung keine Rolle spielt. Der Faktor λ 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 λ 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 Ω als weiteren Bestrafungsterm Γ 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 μ ist ähnlich wie λ 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 Θ und Φ 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 4×4 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.

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

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

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

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 Θ 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 Ψ 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.

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

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

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. Implementierung

Neben der mathematischen Ausarbeitung (siehe Kapitel 5) beschäftigt sich diese Arbeit auch mit der eigentlichen Implementierung des erarbeiteten Verfahrens, für welche folgende grundlegenden Anforderungen bestanden:

Portierbarkeit: In der Aufgabenstellung war bereits gefordert, dass die Implementierung auf verschiedenen Systemen portierbar sein soll. Aus diesem Grund kamen bereits nur einige wenige Programmiersprachen in Frage.

Evaluation der Daten: Da die entstehenden Daten auch evaluiert und grafisch dargestellt werden sollten, war eine weitere Anforderung an die Software, dass sie eine grafische Benutzerschnittstelle (engl. graphical user interface) (GUI) besitzt oder Bild-Formate exportieren kann.

Funktionalität: Die Funktionalität der Implementierung stand primär im Vordergrund. An die Performanz der Implementierung und das Design der GUI wurden keine besonderen Anforderungen gestellt.

Software-Qualität: Da diese Arbeit eine Bachelor-Arbeit der Fachrichtung *Softwaretechnik* ist, bestand eine gewisse Anforderung an die Qualität des Quellcodes. Diese beinhaltet u.a. automatisierte Tests, objektorientierte Programmierung und Modulare Strukturen.

Von einer Laufzeitanalyse des Programmes wurde absichtlich abgesehen. Diese Arbeit ist in erster Linie eine Machbarkeitsstudie und erhebt nicht den Anspruch die gestellten Aufgaben in optimaler Zeit zu lösen.

6.1. Wahl der Programmiersprache

Aus den oben beschriebenen Anforderungen ließen sich insgesamt drei mögliche Programmiersprachen ableiten, aus denen eine gewählt werden musste. In die engere Auswahl kamen dabei C, C# und Java. Aus Tabelle 6.1 geht hervor, dass alle diese drei Programmiersprachen für den Einsatzzweck dieser Arbeit geeignet wären. Keine der Sprachen erfüllt ein Kriterium gar nicht oder sticht in einem Bereich besonders hervor. Lediglich C verliert durch die wenige Unterstützung von Software-Qualitätsmerkmalen (wie objektorientierte Programmierung, Modularität, Klassen, etc.) an Punkten.

Aufgrund meiner in universitären Projekte erlangten Vorkenntnissen in Java, wird die Entwicklung der Software für diese Arbeit in derselben Sprache geschehen.

6. Implementierung

Sprache	Java	C	C#
Portierbarkeit	✓	✓	✓ ^a
GUI	✓	✓ ^b	✓
Objektorientierte Programmierung	✓	✗ ^c	✓
Automatisierte Tests	✓	✓ ^d	✓
Nativer Systemzugriff ^e	✗ ^f	✓	✓

Tabelle 6.1.: Vergleich der Programmiersprachen Java, C und C# mit den Anforderungen.

^aMono (<http://www.mono-project.com>) ist eine portierbare Version von C# die auch auf Unix kompiliert

^bDurch Libraries (z.B. GTK+ <http://www.gtk.org>) können in C auch grafische Benutzeroberflächen entwickelt werden

^cObwohl C keine objektorientierte Sprache ist, ist es grundsätzlich natürlich trotzdem möglich ähnliche Konstrukte zu erzeugen.

^dNicht direkt unterstützt, aber es gibt Testframeworks wie z.B. <http://check.sourceforge.net>

^eDer Zugriff auf native Systemfunktionen ermöglicht häufig einen Performance-Gewinn

^fJava läuft im sog. Java Runtime Environment (JRE) und hat damit nicht direkt Zugriff auf native Funktionen

Für die grafische Evaluation der Ergebnisse wurde in der vorliegende Arbeit eine SWING-Oberfläche, die sowohl die Eingabe der Parameter und Daten sowie die Ausgabe der verschiedenen Diagramme und Bilder ermöglichen soll.

Die Abbildung 6.7 zeigt die einfach gehaltene Oberfläche mit den möglichen Einstellungen.

6.2. Externe Bibliotheken

Für die Implementierung wurden einige bereits bestehenden Komponenten eingesetzt. Diese werden hier kurz beschrieben.

NumericTextField¹

Für die Eingabe der einzelnen Parameter wird eine Implementierung eines numerischen Textfeldes verwendet. Dieses verhindert ungültige Eingaben und erleichtert das Auslesen der numerischen Parameter.

¹<http://www.java2s.com/Code/Java/Swing-JFC/NumericTextField.htm>

metadata-extractor²

Die Bibliothek `metadata-extractor` wird verwendet um automatisiert aus Bilddateien die Belichtungszeit der Bilder auszulesen. Mit dieser wird auch das von Adobe entwickelte `xmp-core3` mit geliefert. Zusammen ermöglichen es die Bibliotheken die Belichtungszeiten der Bilder auszulesen und darzustellen.

JImageChooser⁴

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.

6.3. Architektur

Bei der Architektur der Software wurde eine Model-View-Controller (MVC) Struktur eingehalten. Dieses Pattern 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 möglichst gering gehalten wird [LL10, S. 413].

In allen Klassen wurde grundsätzlich das Prinzip des Information Hiding angewandt. Dadurch kann die innere Arbeit der Klassen und Module möglichst unabhängig ablaufen, was eine Erweiterbarkeit vereinfacht.

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)*

6.3.1. Komponentendiagramm

Die Anwendung lässt sich grundsätzlich in fünf verschiedene Komponenten trennen (siehe Abbildung 6.1).

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.

²<https://code.google.com/p/metadata-extractor/>

³<http://www.adobe.com/devnet/xmp.html>

<http://www.adobe.com/devnet/xmp.html>
<http://docs.oracle.com/javase/tutorial/uiswing/examples/components/index.html#FileChooserDemo2>

6. Implementierung

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 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.

Nachfolgend wird nun auf die einzelnen Komponenten und Klassen näher eingegangen (siehe Abbildung 6.2). 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 6.3 dargestellt.

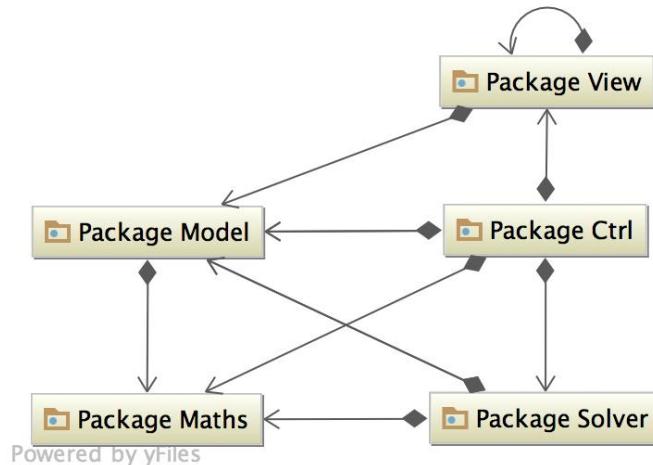


Abbildung 6.1.: Die Architektur der Software auf Komponentenebene. Die Trennung zwischen den einzelnen Bereichen ist hier deutlich erkennbar.

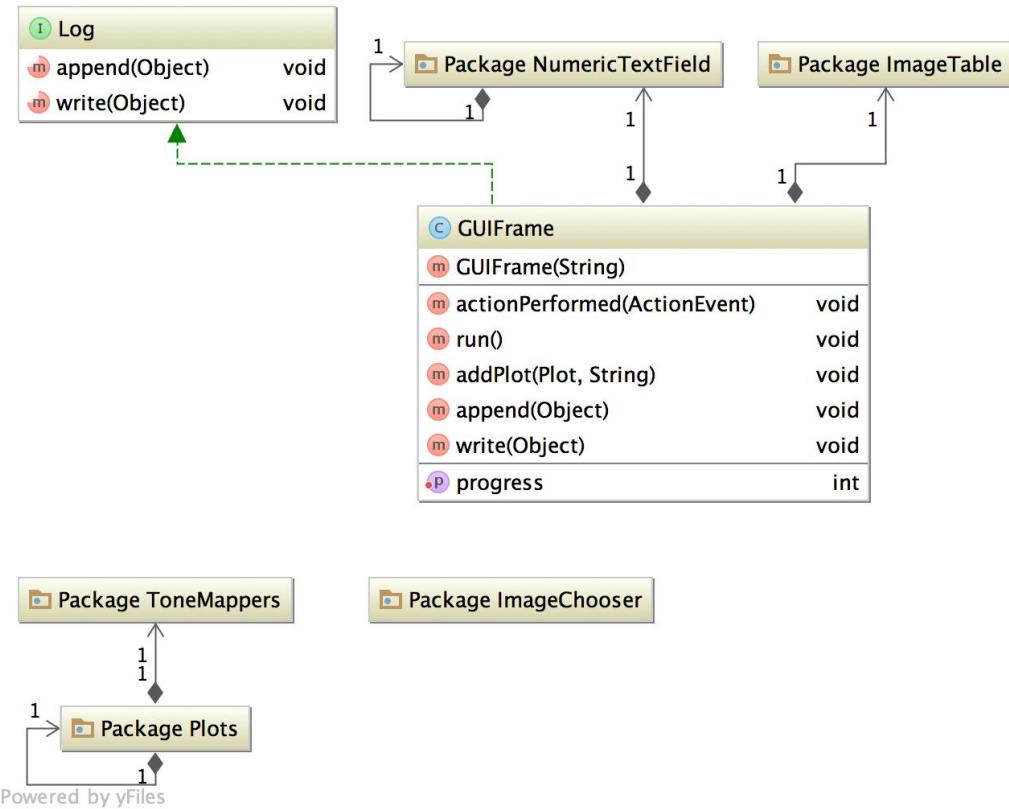


Abbildung 6.2.: Die View-Komponente im Detail. Das Hauptfenster `GUIFrame` stellt die Daten dar und importiert dazu die verschiedenen Komponenten. Die `Plots` und die `ToneMappers` gehören ebenfalls zu diesem Paket.

6. Implementierung

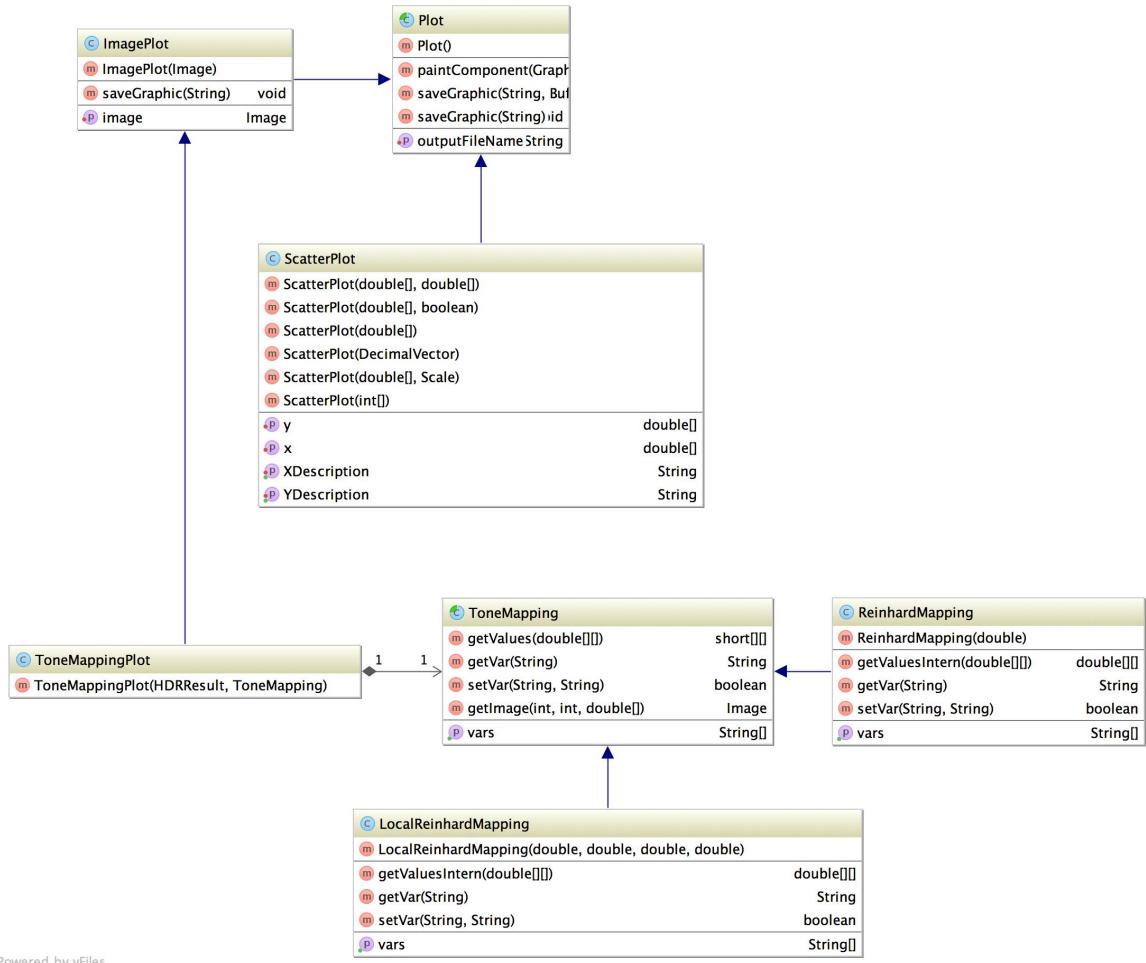


Abbildung 6.3.: Die Struktur der verschiedenen grafischen Plots. Die verschiedenen Tone-Mapping-Operatoren wurden mittels Vererbung implementiert.

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.



Abbildung 6.4.: 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. Implementierung

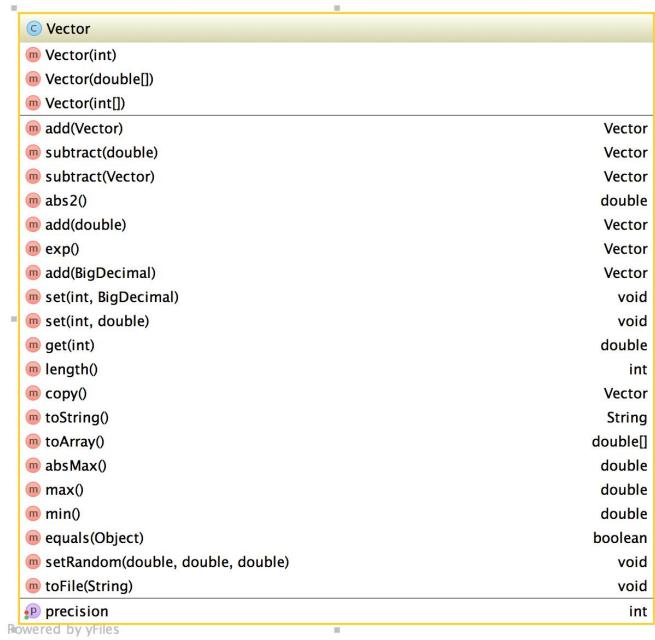


Abbildung 6.5.: Die Klasse `Vector` dient zur Kapselung von Funktionen mit Vektoren und Matrizen.

6.3.2. Sequenzdiagramm

Das Programm läuft sehr vielschichtig ab. Als grundsätzlicher Informationsfluss dient der Aufbau wie er in Abbildung 6.6 beschrieben. Wichtig dabei ist, dass die grafische Benutzeroberfläche nicht von der Berechnungsdauer des Algorithmus blockiert wird, weshalb besonders zwischen Controller und GUIFrame asynchrone Methodenaufrufe eingesetzt werden.

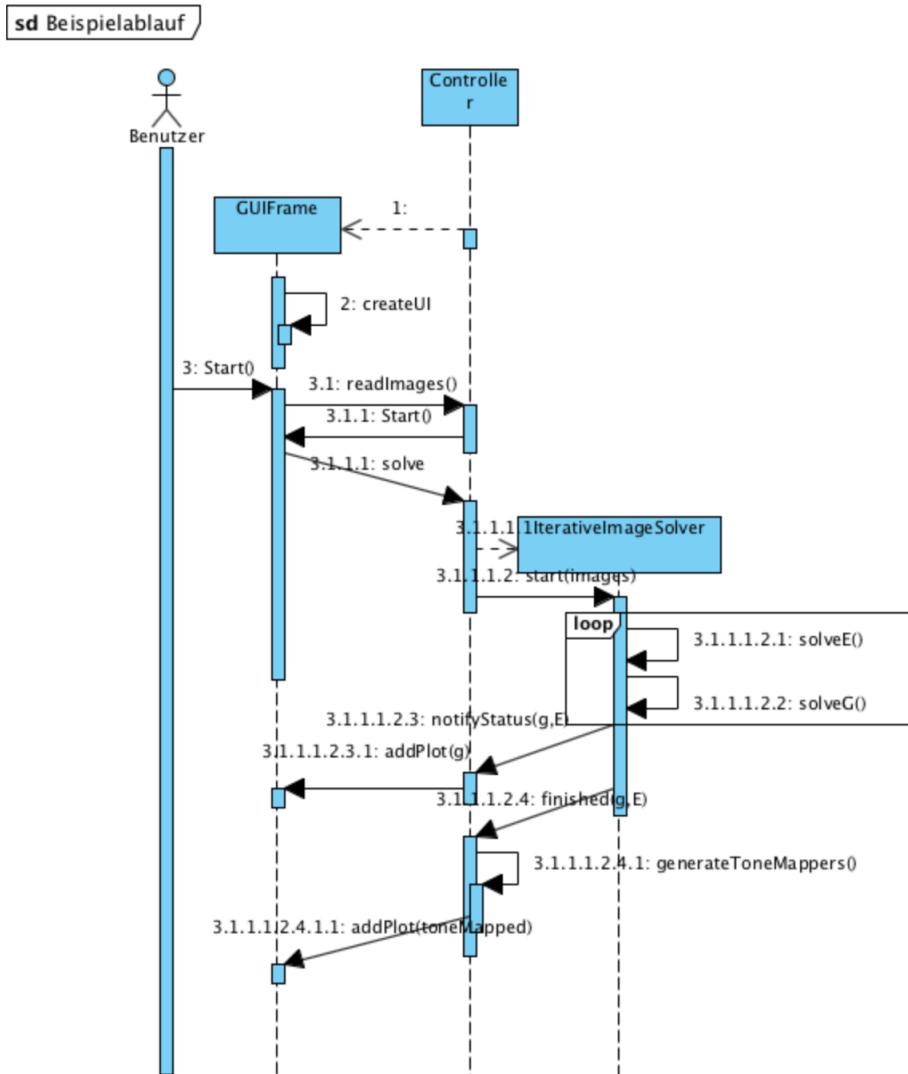


Abbildung 6.6.: Der grundsätzliche Ablauf der Berechnung des HDR-Bildes und der Antwortkurve und der dabei beteiligten Komponenten sowie deren Interaktion.

6.4. Programmvorstellung

Die gesamte implementierte Anwendung befindet sich auf der beigefügten CD-Rom zu dieser Arbeit oder kann unter <https://github.com/sebastianzillessen/hdr-generator> heruntergeladen werden. Das JAR-Archiv kann dann einfach ausgeführt werden.

Die Software unterstützt bisher keine Registrierung der Bilder. D.h. die Bilder müssen vorab von Hand oder mit einem Algorithmus registriert werden und können dann mit diesem Programm verwendet werden.

6. Implementierung

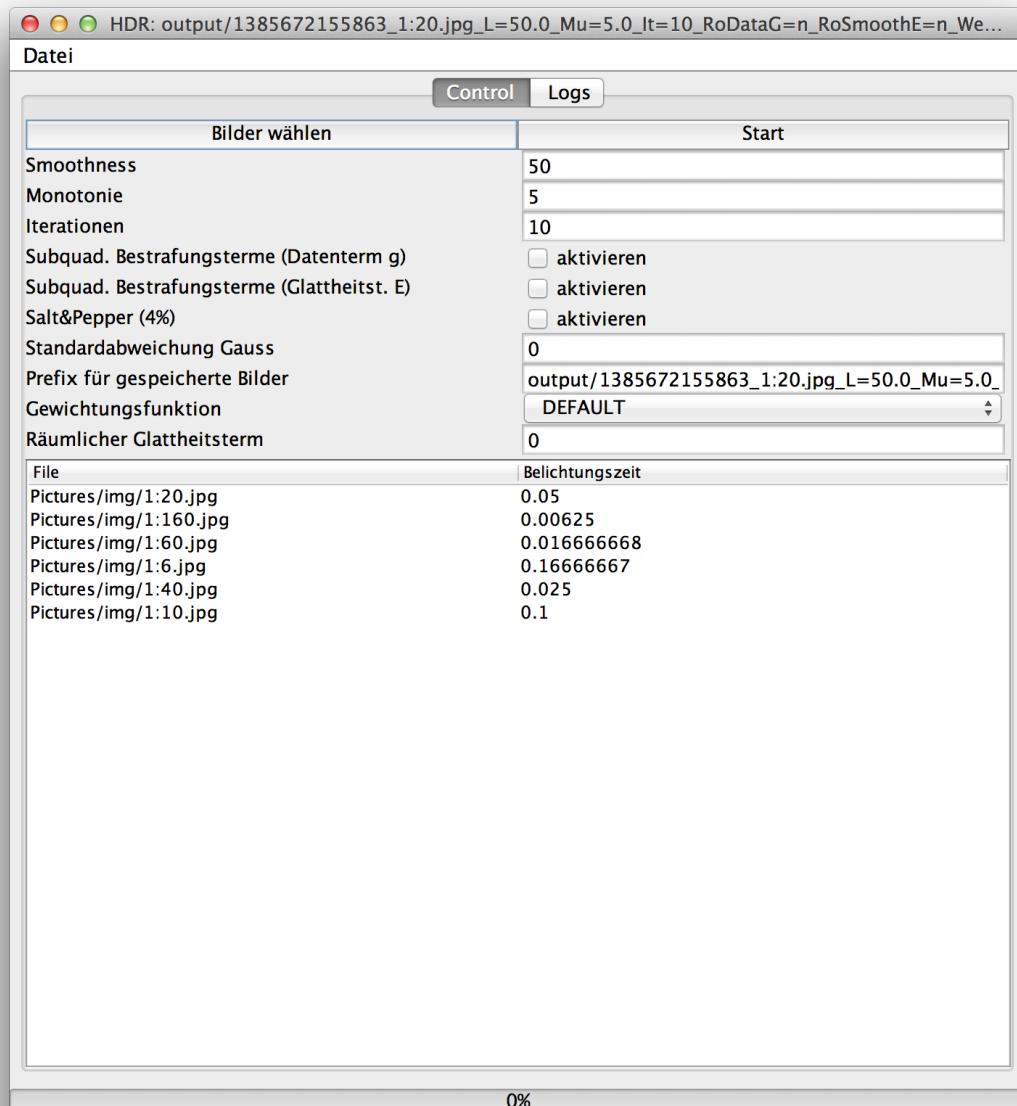


Abbildung 6.7.: Die Benutzereingabe für die Erzeugung der HDR-Bilder ist absichtlich schlicht und einfach gehalten.

6.5. Herausforderungen während der Programmierung

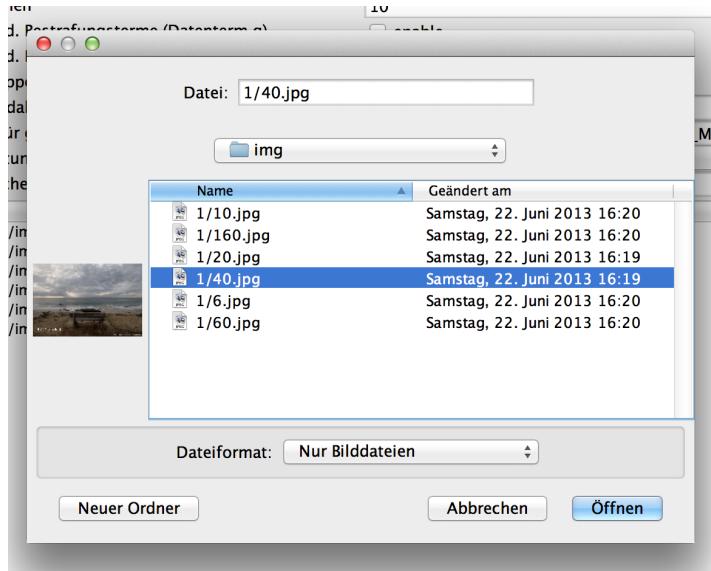


Abbildung 6.8.: Auswahl der Belichtungsserie incl. Vorschau.

Zunächst müssen über den Button „Bilder wählen“ die Eingabebilder gewählt werden (siehe Abbildung 6.8). Sobald die Bilder geladen wurden, erscheinen diese in der unteren Auflistung. Die Belichtungszeit wird in der Tabelle mit angegeben, falls durch die Bibliothek `metadate-extractor` diese Information bereits ausgelesen wurde. Ansonsten müssen diese händisch eingegeben werden.

Die Eingabemöglichkeiten im oberen Bereich des Fensters dienen der Parametrisierung der Anwendung. Hier können insbesondere die Einstellungen zur Monotonie-Forderung, dem räumlichen Glattheitsterm oder den robusten Bestrafungsfunktionen vorgenommen werden.

Außerdem können zu Testzwecken Bildstörungen (Salt & Pepper Rauschen oder additives Gauss-Rauschen) aktiviert werden, wodurch die verschiedenen Verfahren verglichen werden können. Die Ausgabe (auch der Zwischenresultate) erfolgt über die verschiedenen Reiter des Fensters. Darüber hinaus werden die Resultate (globaler und lokaler Reinhard Tone-Mapping-Operator) in diesen ausgegeben (siehe Abbildung 6.9). Der Fortschritt des Verfahrens wird im unteren Bereich der Anwendung dargestellt.

Jeder der verschiedenen Graphen oder Plots kann gespeichert werden. Dazu steht ein Speichern-Button zur Verfügung. Der Funktionsumfang der Anwendung wurde absichtlich schmal gehalten, da der Fokus auf der Erweiterung des Ansatzes und nicht auf der Erstellung einer High-End-Software lag.

6.5. Herausforderungen während der Programmierung

6. Implementierung

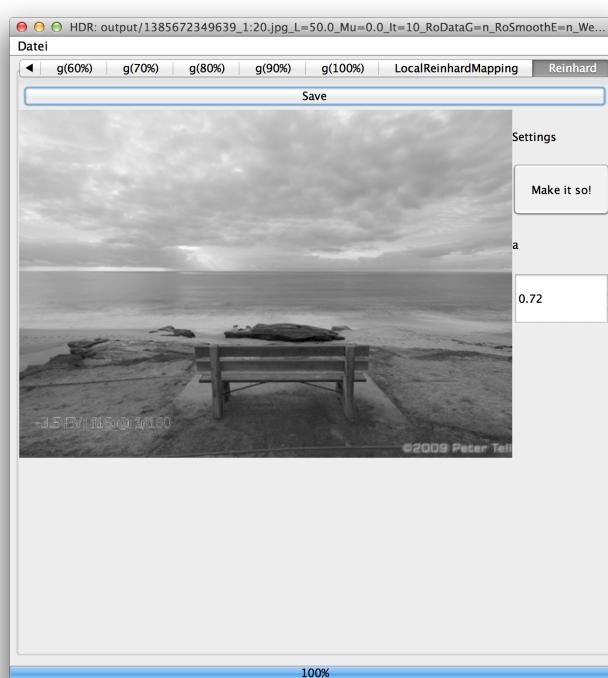


Abbildung 6.9.: Vorschau des Tone-Mapped Resultats mit der Möglichkeit der Veränderung der Variablen für diesen Tone-Mapper (globaler Tone-Mapping-Operator von Reinhard).

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:

λ : Gewicht des Glattheitsterms für g , Standard: 50

μ : Gewicht der Monotonie-Beschränkung für g , Standard: 0

α : Gewicht der räumlichen Glattheit für E , Standard: 0

: Anzahl der Iterationen beim iterativen Lösen, Standard: 10

σ : Standardabweichung des additiven Gauss-Rauschen, Standard: 0

7.1. Ergebnisse mit Erweiterungen

Die in dieser Arbeit vorgestellten Erweiterungen sollen die Resultate noch weiter verbessern. Dazu werden Stück für Stück die Veränderungen in der Antwortkurve gezeigt.

7.1.1. 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 unformige Antwortkurve, welche auch durch mehr Iterationen nicht weiter konvergiert. Durch die Erweiterung der Monotonie kann die Kurve entsprechend begradiert und verbessert werden.

7. Ergebnisse und Resultate

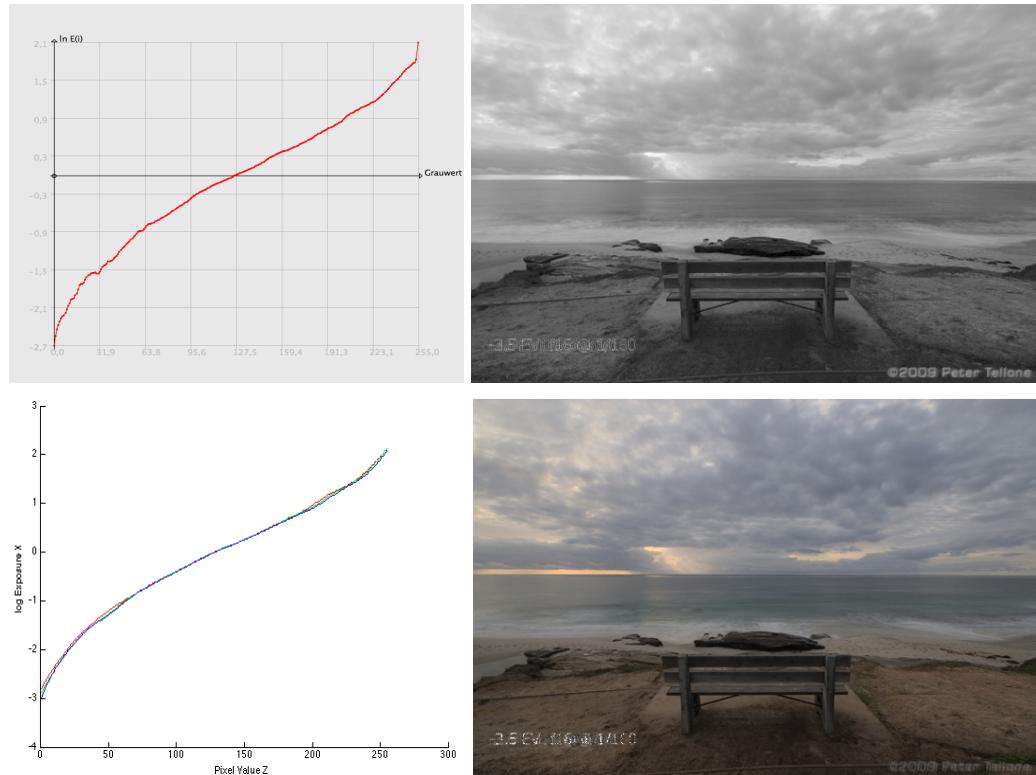


Abbildung 7.1.: Verfahren der Berechnung der Antwortkurve (li.) und dem dazugehörigen Resultat (re.) **oben:** Unser Verfahren, **unten:** Implementierung von Mathias Eitz, siehe Abschnitt 3.1 (keine Erweiterungen aktiviert)

7.1.2. 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 7.4 zu sehen, hat dies insbesondere bei Messfehlern (hier 4% Salt & Pepper Rauschen) einen enormen Vorteil gegenüber dem herkömmlichen Verfahren. Auch durch Störungen wird das additive Gauss-Rauschen ($\sigma = 10$, siehe ??) sind die Ergebnisse schärfer und das Rauschen auf den Eingangsbildern wird reduziert.

7.1.3. Ergebnisse mit Robustheits-Term

Besonders 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) simuliert, dann ist dies im Ausgabebild quasi nicht mehr zu erkennen.

7.1. Ergebnisse mit Erweiterungen

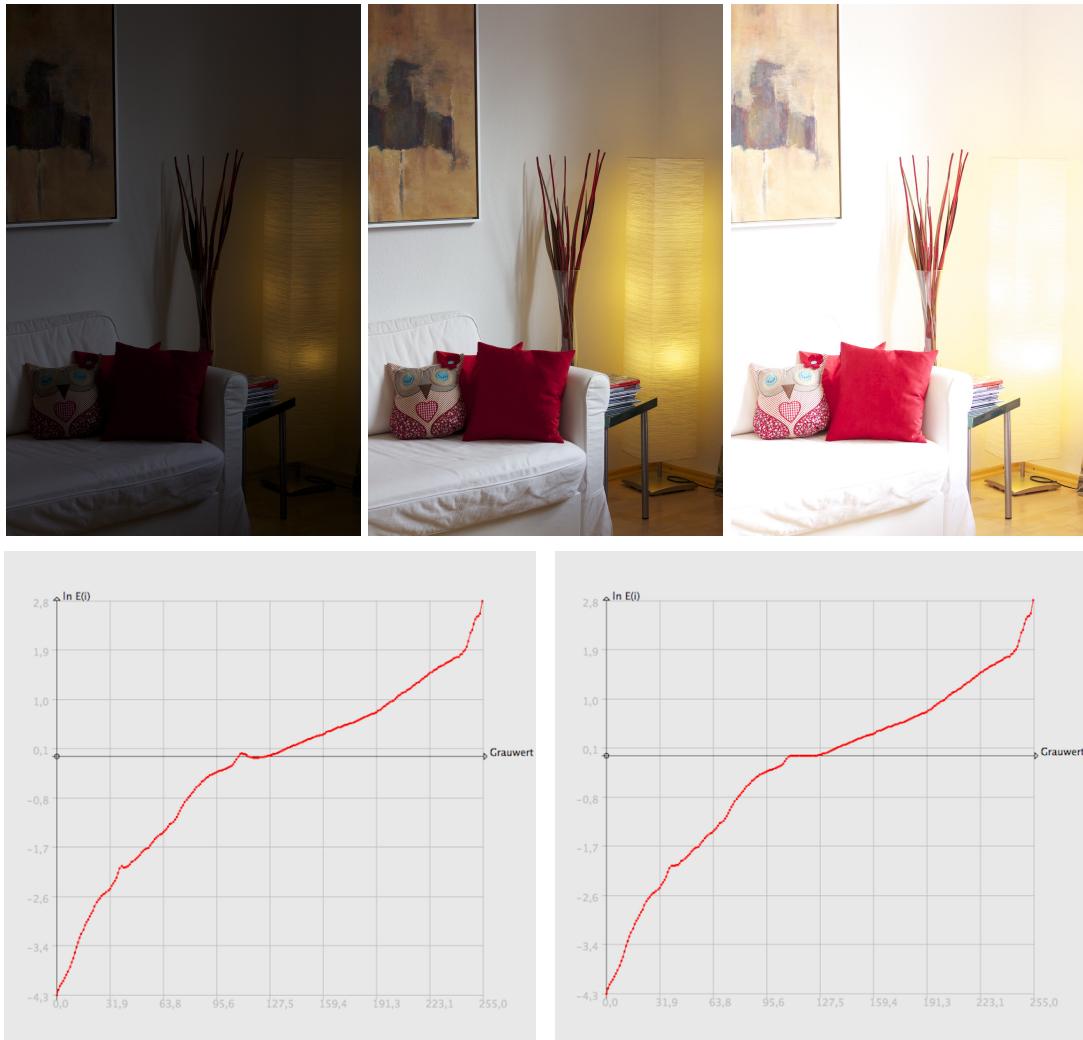


Abbildung 7.2.: oben: Belichtungsserie, Belichtungszeit: 1/8000, 1/640, 1/100 (v.l.), **u.l.:** Antwortkurve ohne Monotonie-Forderung, **u.r.:** Antwortkurve mit Monotonie-Forderung

Insbesondere wirken sich die robusten Bestrafungsfunktionen auch positiv bei der Verwendung der räumlichen Glattheitsforderung aus. Hier fließen die Kanten durch die Verwendung der subquadratischen Terme weniger stark in die räumliche Glattheit ein. Dies sorgt dafür, dass Kanten im Bild besser erhalten bleiben (siehe Abbildung 7.6).

7. Ergebnisse und Resultate

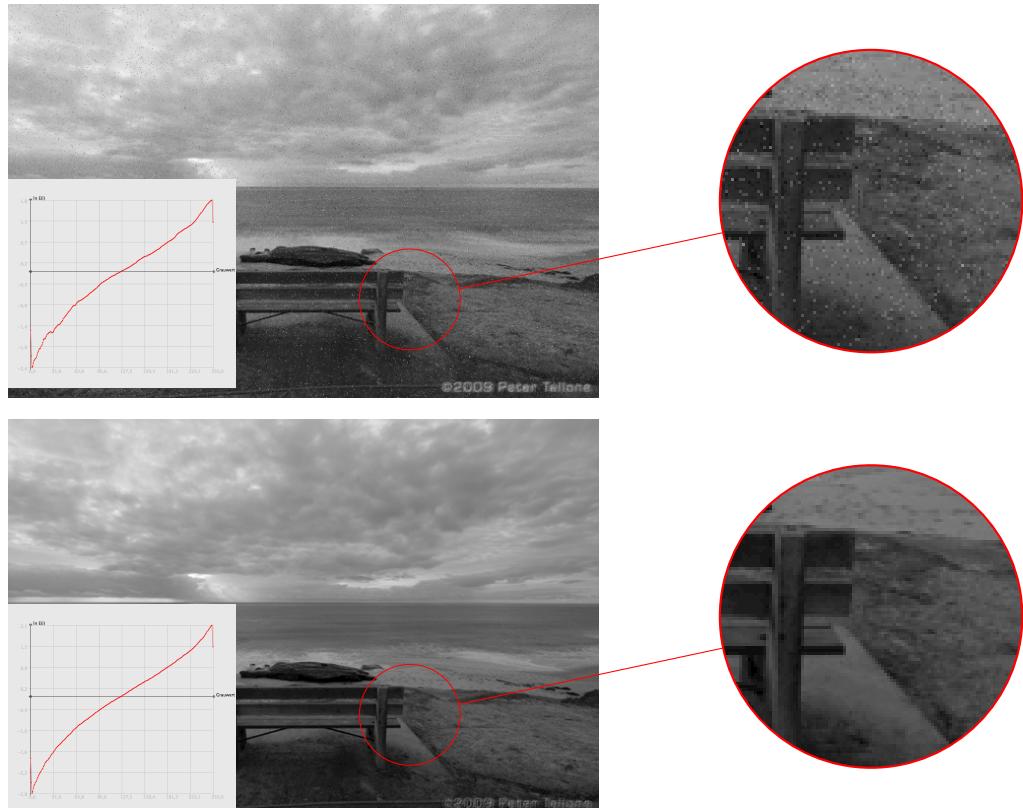


Abbildung 7.3.: Einfluss des räumlichen Glattheitsterms auf verrauschte Bilder (4% Salt & Pepper Rauschen): **oben:** Antwortkurve und zugehöriges HDR Bild ohne räumlichen Glattheitsterm, **unten:** Antwortkurve mit räumlichem Glattheitsterm ($\alpha = 1$) und zugehöriges HDR-Bild. Das Salt & Pepper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

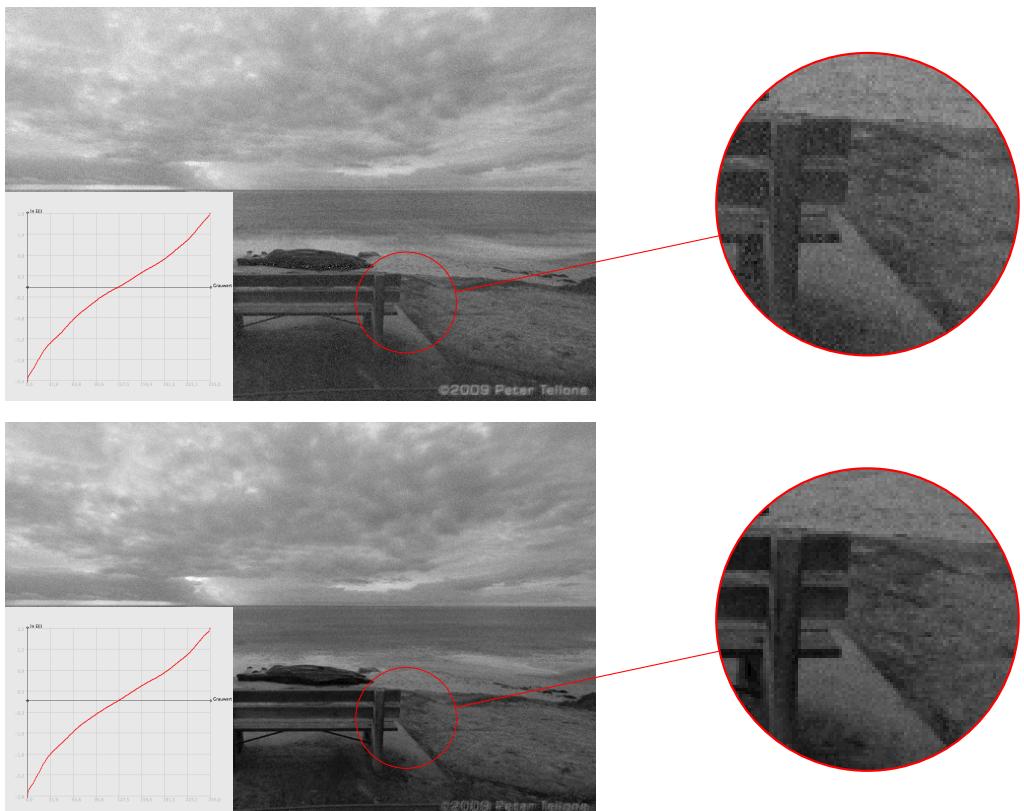


Abbildung 7.4.: Einfluss des räumlichen Glattheitsterms auf verrauschte Bilder (additives Gauss-Rauschen mit $\sigma = 10$): **oben:** Antwortkurve und zugehöriges HDR Bild ohne räumlichen Glattheitsterm, **unten:** Antwortkurve mit räumlichem Glattheitsterm ($\alpha = 1$) und zugehöriges HDR-Bild.

7. Ergebnisse und Resultate

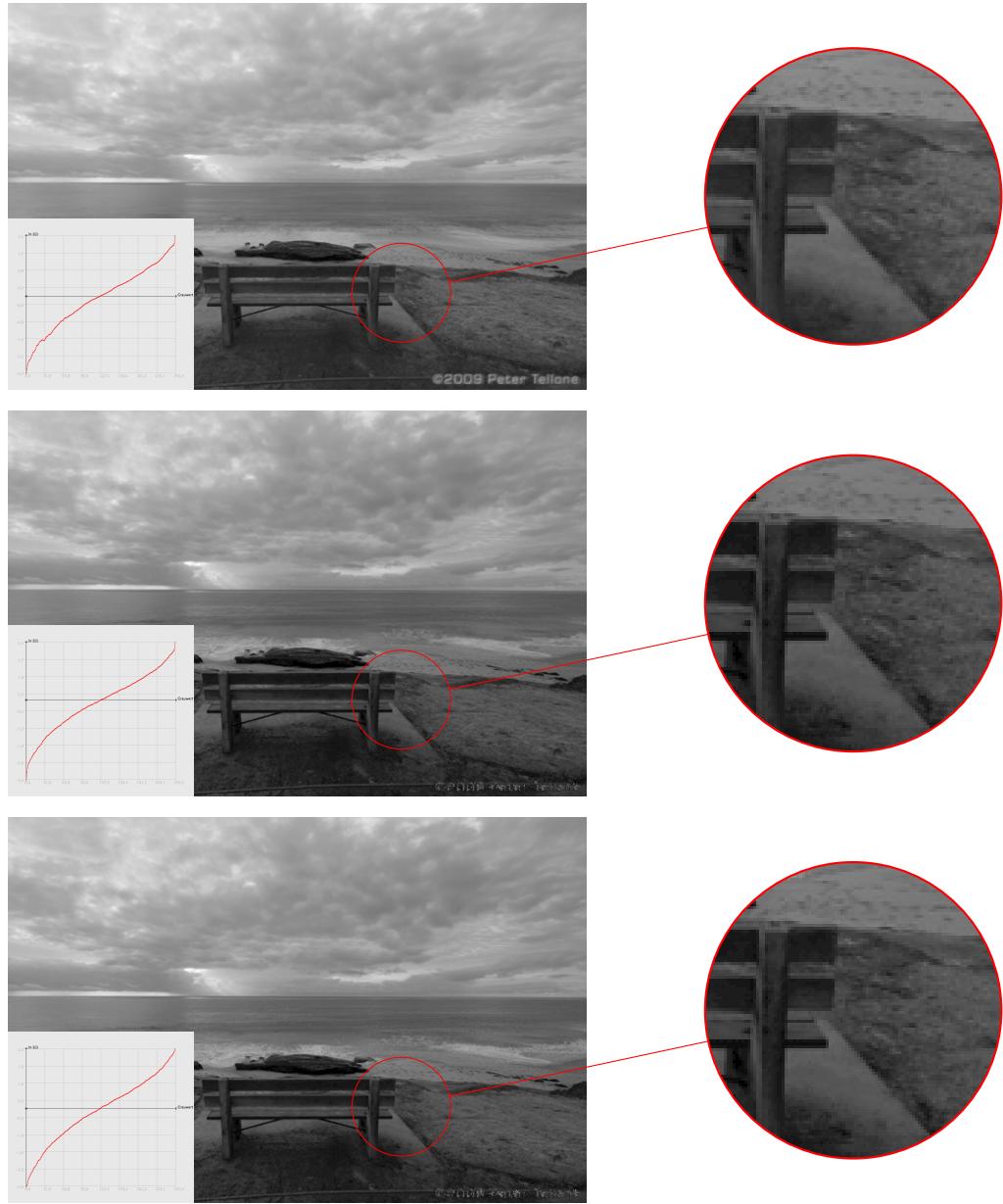


Abbildung 7.5.: Einfluss der subquadratischen Bestrafungsfunktionen: **oben:** Standardverfahren, **mitte:** Subquadratischer Bestrafungsterm $\varphi(s^2) = \sqrt{s^2 + \epsilon^2}$ für \mathbf{g} und \mathbf{E} , die Kanten sind schärfer und der Kontrast besser, **unten:** Zusätzliches Salt & Pepper Rauschen ist quasi nicht mehr zu erkennen.

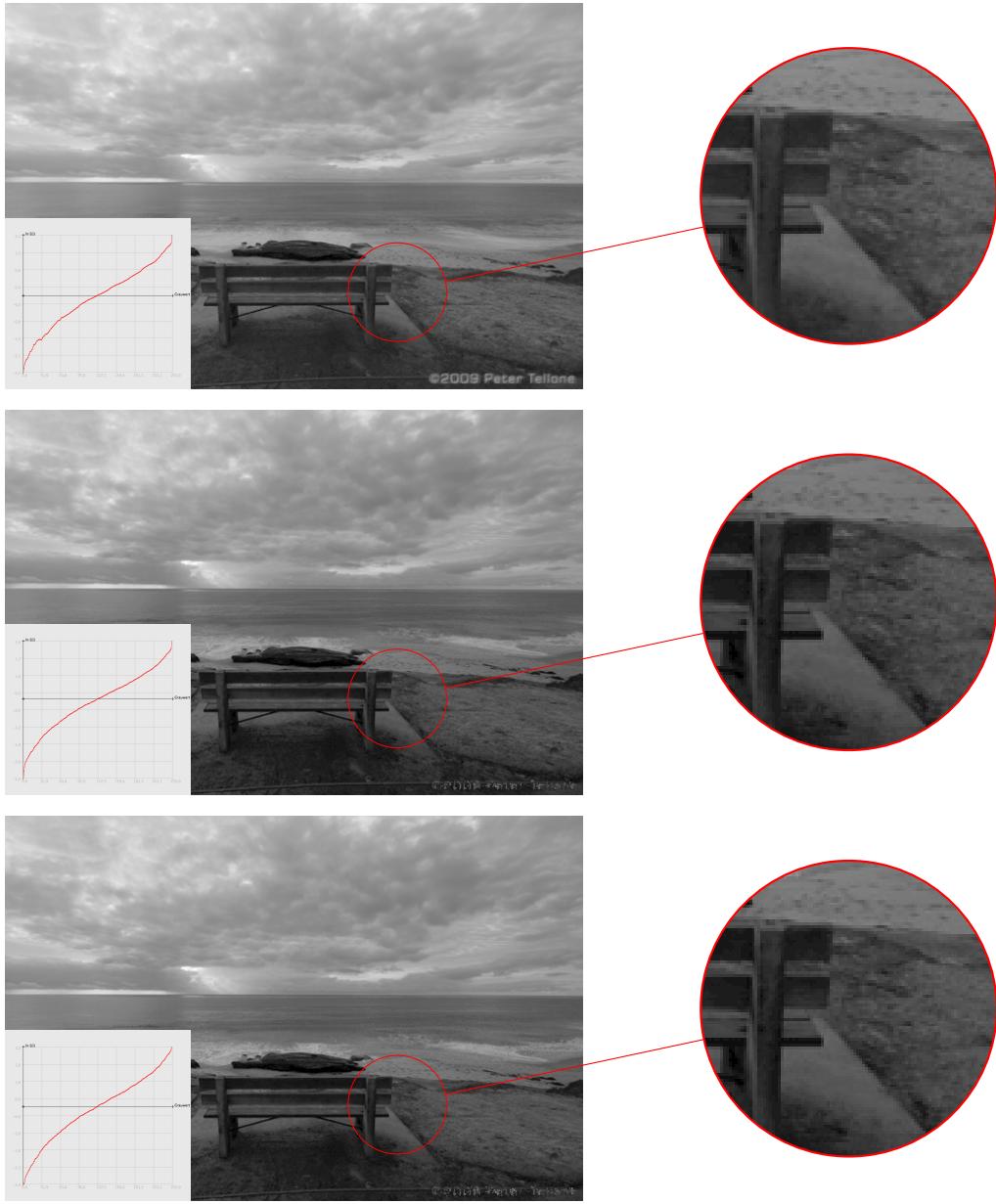


Abbildung 7.6.: Einfluss der subquadratischen Bestrafungsfunktionen auf den räumlichen Glattheitsterm: **oben:** Standardverfahren, **mitte:** Subquadratischer Bestrafungsterm $\varphi(s^2) = \sqrt{s^2 + \epsilon^2}$ für \mathbf{g} und \mathbf{E} , die Kanten sind schärfer und der Kontrast besser, **unten:** Zusätzliches Salt & Pepper Rauschen ist quasi nicht mehr zu erkennen.

8. Zusammenfassung und Ausblick

Hier bitte einen kurzen Durchgang durch die Arbeit.

Ausblick

...und anschließend einen Ausblick

A. Anhang

A.1. Artikel [DM97]

Recovering High Dynamic Range Radiance Maps from Photographs

Paul E. Debevec

Jitendra Malik

University of California at Berkeley¹

ABSTRACT

We present a method of recovering high dynamic range radiance maps from photographs taken with conventional imaging equipment. In our method, multiple photographs of the scene are taken with different amounts of exposure. Our algorithm uses these differently exposed photographs to recover the response function of the imaging process, up to factor of scale, using the assumption of reciprocity. With the known response function, the algorithm can fuse the multiple photographs into a single, high dynamic range radiance map whose pixel values are proportional to the true radiance values in the scene. We demonstrate our method on images acquired with both photochemical and digital imaging processes. We discuss how this work is applicable in many areas of computer graphics involving digitized photographs, including image-based modeling, image compositing, and image processing. Lastly, we demonstrate a few applications of having high dynamic range radiance maps, such as synthesizing realistic motion blur and simulating the response of the human visual system.

CR Descriptors: I.2.10 [Artificial Intelligence]: Vision and Scene Understanding - *Intensity, color, photometry and thresholding*; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism - *Color, shading, shadowing, and texture*; I.4.1 [Image Processing]: Digitization - *Scanning*; I.4.8 [Image Processing]: Scene Analysis - *Photometry, Sensor Fusion*.

1 Introduction

Digitized photographs are becoming increasingly important in computer graphics. More than ever, scanned images are used as texture maps for geometric models, and recent work in image-based modeling and rendering uses images as the fundamental modeling primitive. Furthermore, many of today's graphics applications require computer-generated images to mesh seamlessly with real photographic imagery. Properly using photographically acquired imagery in these applications can greatly benefit from an accurate model of the photographic process.

When we photograph a scene, either with film or an electronic imaging array, and digitize the photograph to obtain a two-dimensional array of "brightness" values, these values are rarely

true measurements of relative radiance in the scene. For example, if one pixel has twice the value of another, it is unlikely that it observed twice the radiance. Instead, there is usually an unknown, nonlinear mapping that determines how radiance in the scene becomes pixel values in the image.

This nonlinear mapping is hard to know beforehand because it is actually the composition of several nonlinear mappings that occur in the photographic process. In a conventional camera (see Fig. 1), the film is first exposed to light to form a latent image. The film is then developed to change this latent image into variations in transparency, or *density*, on the film. The film can then be digitized using a film scanner, which projects light through the film onto an electronic light-sensitive array, converting the image to electrical voltages. These voltages are digitized, and then manipulated before finally being written to the storage medium. If prints of the film are scanned rather than the film itself, then the printing process can also introduce nonlinear mappings.

In the first stage of the process, the film response to variations in exposure X (which is $E\Delta t$, the product of the irradiance E the film receives and the exposure time Δt) is a non-linear function, called the "characteristic curve" of the film. Noteworthy in the typical characteristic curve is the presence of a small response with no exposure and saturation at high exposures. The development, scanning and digitization processes usually introduce their own nonlinearities which compose to give the aggregate nonlinear relationship between the image pixel exposures X and their values Z .

Digital cameras, which use charge coupled device (CCD) arrays to image the scene, are prone to the same difficulties. Although the charge collected by a CCD element is proportional to its irradiance, most digital cameras apply a nonlinear mapping to the CCD outputs before they are written to the storage medium. This nonlinear mapping is used in various ways to mimic the response characteristics of film, anticipate nonlinear responses in the display device, and often to convert 12-bit output from the CCD's analog-to-digital converters to 8-bit values commonly used to store images. As with film, the most significant nonlinearity in the response curve is at its saturation point, where any pixel with a radiance above a certain level is mapped to the same maximum image value.

Why is this any problem at all? The most obvious difficulty, as any amateur or professional photographer knows, is that of limited dynamic range—one has to choose the range of radiance values that are of interest and determine the exposure time suitably. Sunlit scenes, and scenes with shiny materials and artificial light sources, often have extreme differences in radiance values that are impossible to capture without either under-exposing or saturating the film. To cover the full dynamic range in such a scene, one can take a series of photographs with different exposures. This then poses a problem: how can we combine these separate images into a composite radiance map? Here the fact that the mapping from scene radiance to pixel values is unknown and nonlinear begins to haunt us. The purpose of this paper is to present a simple technique for recovering this response function, up to a scale factor, using nothing more than a set of photographs taken with varying, known exposure durations. With this mapping, we then use the pixel values from all available photographs to construct an accurate map of the radiance in the scene, up to a factor of scale. This radiance map will cover

¹Computer Science Division, University of California at Berkeley, Berkeley, CA 94720-1776. Email: debevec@cs.berkeley.edu, malik@cs.berkeley.edu. More information and additional results may be found at: <http://www.cs.berkeley.edu/~debevec/Research>

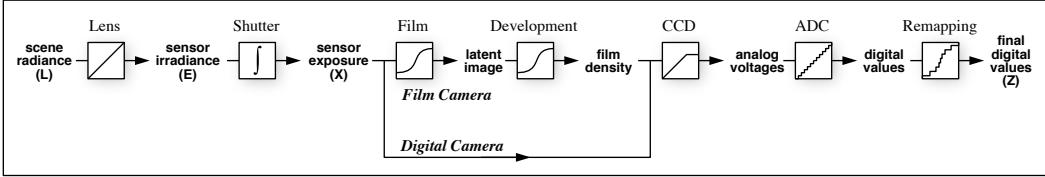


Figure 1: **Image Acquisition Pipeline** shows how scene radiance becomes pixel values for both film and digital cameras. Unknown nonlinear mappings can occur during exposure, development, scanning, digitization, and remapping. The algorithm in this paper determines the aggregate mapping from scene radiance L to pixel values Z from a set of differently exposed images.

the entire dynamic range captured by the original photographs.

1.1 Applications

Our technique of deriving imaging response functions and recovering high dynamic range radiance maps has many possible applications in computer graphics:

Image-based modeling and rendering

Image-based modeling and rendering systems to date (e.g. [11, 15, 2, 3, 12, 6, 17]) make the assumption that all the images are taken with the same exposure settings and film response functions. However, almost any large-scale environment will have some areas that are much brighter than others, making it impossible to adequately photograph the scene using a single exposure setting. In indoor scenes with windows, this situation often arises within the field of view of a single photograph, since the areas visible through the windows can be far brighter than the areas inside the building.

By determining the response functions of the imaging device, the method presented here allows one to correctly fuse pixel data from photographs taken at different exposure settings. As a result, one can properly photograph outdoor areas with short exposures, and indoor areas with longer exposures, without creating inconsistencies in the data set. Furthermore, knowing the response functions can be helpful in merging photographs taken with different imaging systems, such as video cameras, digital cameras, and film cameras with various film stocks and digitization processes.

The area of image-based modeling and rendering is working toward recovering more advanced reflection models (up to complete BRDF's) of the surfaces in the scene (e.g. [21]). These methods, which involve observing surface radiance in various directions under various lighting conditions, require absolute radiance values rather than the nonlinearly mapped pixel values found in conventional images. Just as important, the recovery of high dynamic range images will allow these methods to obtain accurate radiance values from surface specularities and from incident light sources. Such higher radiance values usually become clamped in conventional images.

Image processing

Most image processing operations, such as blurring, edge detection, color correction, and image correspondence, expect pixel values to be proportional to the scene radiance. Because of nonlinear image response, especially at the point of saturation, these operations can produce incorrect results for conventional images.

In computer graphics, one common image processing operation is the application of synthetic motion blur to images. In our results (Section 3), we will show that using true radiance maps produces significantly more realistic motion blur effects for high dynamic range scenes.

Image compositing

Many applications in computer graphics involve compositing image data from images obtained by different processes. For example, a background matte might be shot with a still camera, live action might be shot with a different film stock or scanning process, and CG elements would be produced by rendering algorithms. When there are significant differences in the response curves of these imaging processes, the composite image can be visually unconvincing. The technique presented in this paper provides a convenient and robust method of determining the overall response curve of any imaging process, allowing images from different processes to be used consistently as radiance maps. Furthermore, the recovered response curves can be inverted to render the composite radiance map as if it had been photographed with any of the original imaging processes, or a different imaging process entirely.

A research tool

One goal of computer graphics is to simulate the image formation process in a way that produces results that are consistent with what happens in the real world. Recovering radiance maps of real-world scenes should allow more quantitative evaluations of rendering algorithms to be made in addition to the qualitative scrutiny they traditionally receive. In particular, the method should be useful for developing reflectance and illumination models, and comparing global illumination solutions against ground truth data.

Rendering high dynamic range scenes on conventional display devices is the subject of considerable previous work, including [20, 16, 5, 23]. The work presented in this paper will allow such methods to be tested on real radiance maps in addition to synthetically computed radiance solutions.

1.2 Background

The photochemical processes involved in silver halide photography have been the subject of continued innovation and research ever since the invention of the daguerreotype in 1839. [18] and [8] provide a comprehensive treatment of the theory and mechanisms involved. For the newer technology of solid-state imaging with charge coupled devices, [19] is an excellent reference. The technical and artistic problem of representing the dynamic range of a natural scene on the limited range of film has concerned photographers from the early days – [1] presents one of the best known systems to choose shutter speeds, lens apertures, and developing conditions to best coerce the dynamic range of a scene to fit into what is possible on a print. In scientific applications of photography, such as in astronomy, the nonlinear film response has been addressed by suitable calibration procedures. It is our objective instead to develop a simple self-calibrating procedure not requiring calibration charts or photometric measuring devices.

In previous work, [13] used multiple flux integration times of a CCD array to acquire extended dynamic range images. Since direct CCD outputs were available, the work did not need to deal with the

A. Anhang

problem of nonlinear pixel value response. [14] addressed the problem of nonlinear response but provide a rather limited method of recovering the response curve. Specifically, a parametric form of the response curve is arbitrarily assumed, there is no satisfactory treatment of image noise, and the recovery process makes only partial use of the available data.

2 The Algorithm

This section presents our algorithm for recovering the film response function, and then presents our method of reconstructing the high dynamic range radiance image from the multiple photographs. We describe the algorithm assuming a grayscale imaging device. We discuss how to deal with color in Section 2.6.

2.1 Film Response Recovery

Our algorithm is based on exploiting a physical property of imaging systems, both photochemical and electronic, known as *reciprocity*.

Let us consider photographic film first. The response of a film to variations in exposure is summarized by the characteristic curve (or Hurter-Driffield curve). This is a graph of the optical density D of the processed film against the logarithm of the exposure \bar{X} to which it has been subjected. The exposure X is defined as the product of the irradiance E at the film and exposure time, Δt , so that its units are J m^{-2} . Key to the very concept of the characteristic curve is the assumption that only the product $E\Delta t$ is important, that halving E and doubling Δt will not change the resulting optical density D . Under extreme conditions (very large or very low Δt), the reciprocity assumption can break down, a situation described as reciprocity failure. In typical print films, reciprocity holds to within $\frac{1}{3}$ stop¹ for exposure times of 10 seconds to 1/10,000 of a second.² In the case of charge coupled arrays, reciprocity holds under the assumption that each site measures the total number of photons it absorbs during the integration time.

After the development, scanning and digitization processes, we obtain a digital number Z , which is a nonlinear function of the original exposure X at the pixel. Let us call this function f , which is the composition of the characteristic curve of the film as well as all the nonlinearities introduced by the later processing steps. Our first goal will be to recover this function f . Once we have that, we can compute the exposure X at each pixel, as $X = f^{-1}(Z)$. We make the reasonable assumption that the function f is monotonically increasing, so its inverse f^{-1} is well defined. Knowing the exposure X and the exposure time Δt , the irradiance E is recovered as $E = X/\Delta t$, which we will take to be proportional to the radiance L in the scene.³

Before proceeding further, we should discuss the consequences of the spectral response of the sensor. The exposure X should be thought of as a function of wavelength $X(\lambda)$, and the abscissa on the characteristic curve should be the integral $\int X(\lambda)R(\lambda)d\lambda$ where $R(\lambda)$ is the spectral response of the sensing element at the pixel location. Strictly speaking, our use of irradiance, a radiometric quantity, is not justified. However, the spectral response of the sensor site may not be the photopic luminosity function V_λ , so the photometric term *illuminance* is not justified either. In what follows, we will use the term irradiance, while urging the reader to remember that the

¹1 stop is a photographic term for a factor of two; $\frac{1}{3}$ stop is thus $2^{\frac{1}{3}}$

²An even larger dynamic range can be covered by using neutral density filters to lessen to amount of light reaching the film for a given exposure time. A discussion of the modes of reciprocity failure may be found in [18], ch. 4.

³ L is proportional E for any particular pixel, but it is possible for the proportionality factor to be different at different places on the sensor. One formula for this variance, given in [7], is $E = L \frac{\pi}{4} \left(\frac{d}{f} \right)^2 \cos^4 \alpha$, where α measures the pixel's angle from the lens' optical axis. However, most modern camera lenses are designed to compensate for this effect, and provide a nearly constant mapping between radiance and irradiance at f/8 and smaller apertures. See also [10].

quantities we will be dealing with are weighted by the spectral response at the sensor site. For color photography, the color channels may be treated separately.

The input to our algorithm is a number of digitized photographs taken from the same vantage point with different known exposure durations Δt_j .⁴ We will assume that the scene is static and that this process is completed quickly enough that lighting changes can be safely ignored. It can then be assumed that the film irradiance values E_i for each pixel i are constant. We will denote pixel values by Z_{ij} where i is a spatial index over pixels and j indexes over exposure times Δt_j . We may now write down the film reciprocity equation as:

$$Z_{ij} = f(E_i \Delta t_j) \quad (1)$$

Since we assume f is monotonic, it is invertible, and we can rewrite (1) as:

$$f^{-1}(Z_{ij}) = E_i \Delta t_j$$

Taking the natural logarithm of both sides, we have:

$$\ln f^{-1}(Z_{ij}) = \ln E_i + \ln \Delta t_j$$

To simplify notation, let us define function $g = \ln f^{-1}$. We then have the set of equations:

$$g(Z_{ij}) = \ln E_i + \ln \Delta t_j \quad (2)$$

where i ranges over pixels and j ranges over exposure durations. In this set of equations, the Z_{ij} are known, as are the Δt_j . The unknowns are the irradiances E_i , as well as the function g , although we assume that g is smooth and monotonic.

We wish to recover the function g and the irradiances E_i that best satisfy the set of equations arising from Equation 2 in a least-squared error sense. We note that recovering g only requires recovering the *finite* number of values that $g(z)$ can take since the domain of Z , pixel brightness values, is finite. Letting Z_{min} and Z_{max} be the least and greatest pixel values (integers), N be the number of pixel locations and P be the number of photographs, we formulate the problem as one of finding the $(Z_{max} - Z_{min} + 1)$ values of $g(Z)$ and the N values of $\ln E_i$ that minimize the following quadratic objective function:

$$\mathcal{O} = \sum_{i=1}^N \sum_{j=1}^P [g(Z_{ij}) - \ln E_i - \ln \Delta t_j]^2 + \lambda \sum_{z=Z_{min}+1}^{Z_{max}-1} g''(z)^2 \quad (3)$$

The first term ensures that the solution satisfies the set of equations arising from Equation 2 in a least squares sense. The second term is a smoothness term on the sum of squared values of the second derivative of g to ensure that the function g is smooth; in this discrete setting we use $g''(z) = g(z-1) - 2g(z) + g(z+1)$. This smoothness term is essential to the formulation in that it provides coupling between the values $g(z)$ in the minimization. The scalar λ weights the smoothness term relative to the data fitting term, and should be chosen appropriately for the amount of noise expected in the Z_{ij} measurements.

Because it is quadratic in the E_i 's and $g(z)$'s, minimizing \mathcal{O} is a straightforward linear least squares problem. The overdetermined

⁴Most modern SLR cameras have electronically controlled shutters which give extremely accurate and reproducible exposure times. We tested our Canon EOS Elan camera by using a Macintosh to make digital audio recordings of the shutter. By analyzing these recordings we were able to verify the accuracy of the exposure times to within a thousandth of a second. Conveniently, we determined that the actual exposure times varied by powers of two between stops ($\frac{1}{64}, \frac{1}{32}, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 2, 4, 8, 16, 32$), rather than the rounded numbers displayed on the camera readout ($\frac{1}{60}, \frac{1}{30}, \frac{1}{15}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 2, 4, 8, 15, 30$). Because of problems associated with vignetting, varying the aperture is not recommended.

system of linear equations is robustly solved using the singular value decomposition (SVD) method. An intuitive explanation of the procedure may be found in Fig. 2.

We need to make three additional points to complete our description of the algorithm:

First, the solution for the $g(z)$ and E_i values can only be up to a single scale factor α . If each log irradiance value $\ln E_i$ were replaced by $\ln E_i + \alpha$, and the function g replaced by $g + \alpha$, the system of equations 2 and also the objective function \mathcal{O} would remain unchanged. To establish a scale factor, we introduce the additional constraint $g(Z_{mid}) = 0$, where $Z_{mid} = \frac{1}{2}(Z_{min} + Z_{max})$, simply by adding this as an equation in the linear system. The meaning of this constraint is that a pixel with value midway between Z_{min} and Z_{max} will be assumed to have unit exposure.

Second, the solution can be made to have a much better fit by anticipating the basic shape of the response function. Since $g(z)$ will typically have a steep slope near Z_{min} and Z_{max} , we should expect that $g(z)$ will be less smooth and will fit the data more poorly near these extremes. To recognize this, we can introduce a weighting function $w(z)$ to emphasize the smoothness and fitting terms toward the middle of the curve. A sensible choice of w is a simple hat function:

$$w(z) = \begin{cases} z - Z_{min} & \text{for } z \leq \frac{1}{2}(Z_{min} + Z_{max}) \\ Z_{max} - z & \text{for } z > \frac{1}{2}(Z_{min} + Z_{max}) \end{cases} \quad (4)$$

Equation 3 now becomes:

$$\begin{aligned} \mathcal{O} = & \sum_{i=1}^N \sum_{j=1}^P \{w(Z_{ij})[g(Z_{ij}) - \ln E_i - \ln \Delta t_j]\}^2 + \\ & \lambda \sum_{z=Z_{min}+1}^{Z_{max}-1} [w(z)g''(z)]^2 \end{aligned}$$

Finally, we need not use every available pixel site in this solution procedure. Given measurements of N pixels in P photographs, we have to solve for N values of $\ln E_i$ and $(Z_{max} - Z_{min})$ samples of g . To ensure a sufficiently overdetermined system, we want $N(P-1) > (Z_{max} - Z_{min})$. For the pixel value range $(Z_{max} - Z_{min}) = 255$, $P = 11$ photographs, a choice of N on the order of 50 pixels is more than adequate. Since the size of the system of linear equations arising from Equation 3 is on the order of $N \times P + Z_{max} - Z_{min}$, computational complexity considerations make it impractical to use every pixel location in this algorithm. Clearly, the pixel locations should be chosen so that they have a reasonably even distribution of pixel values from Z_{min} to Z_{max} , and so that they are spatially well distributed in the image. Furthermore, the pixels are best sampled from regions of the image with low intensity variance so that radiance can be assumed to be constant across the area of the pixel, and the effect of optical blur of the imaging system is minimized. So far we have performed this task by hand, though it could easily be automated.

Note that we have not explicitly enforced the constraint that g must be a monotonic function. If desired, this can be done by transforming the problem to a non-negative least squares problem. We have not found it necessary because, in our experience, the smoothness penalty term is enough to make the estimated g monotonic in addition to being smooth.

To show its simplicity, the MATLAB routine we used to minimize Equation 5 is included in the Appendix. Running times are on the order of a few seconds.

2.2 Constructing the High Dynamic Range Radiance Map

Once the response curve g is recovered, it can be used to quickly convert pixel values to relative radiance values, assuming the exposure Δt_j is known. Note that the curve can be used to determine radiance values in any image(s) acquired by the imaging process associated with g , not just the images used to recover the response function.

From Equation 2, we obtain:

$$\ln E_i = g(Z_{ij}) - \ln \Delta t_j \quad (5)$$

For robustness, and to recover high dynamic range radiance values, we should use all the available exposures for a particular pixel to compute its radiance. For this, we reuse the weighting function in Equation 4 to give higher weight to exposures in which the pixel's value is closer to the middle of the response function:

$$\ln E_i = \frac{\sum_{j=1}^P w(Z_{ij})(g(Z_{ij}) - \ln \Delta t_j)}{\sum_{j=1}^P w(Z_{ij})} \quad (6)$$

Combining the multiple exposures has the effect of reducing noise in the recovered radiance values. It also reduces the effects of imaging artifacts such as film grain. Since the weighting function ignores saturated pixel values, "blooming" artifacts⁵ have little impact on the reconstructed radiance values.

2.2.1 Storage

In our implementation the recovered radiance map is computed as an array of single-precision floating point values. For efficiency, the map can be converted to the image format used in the RADIANCE [22] simulation and rendering system, which uses just eight bits for each of the mantissa and exponent. This format is particularly compact for color radiance maps, since it stores just one exponent value for all three color values at each pixel. Thus, in this format, a high dynamic range radiance map requires just one third more storage than a conventional RGB image.

2.3 How many images are necessary?

To decide on the number of images needed for the technique, it is convenient to consider the two aspects of the process:

- Recovering the film response curve:* This requires a minimum of two photographs. Whether two photographs are enough can be understood in terms of the heuristic explanation of the process of film response curve recovery shown in Fig. 2. If the scene has sufficiently many different radiance values, the entire curve can, in principle, be assembled by sliding together the sampled curve segments, each with only two samples. Note that the photos must be similar enough in their exposure amounts that some pixels fall into the working range⁶ of the film in both images; otherwise, there is no information to relate the exposures to each other. Obviously, using more than two images with differing exposure times improves performance with respect to noise sensitivity.
- Recovering a radiance map given the film response curve:* The number of photographs needed here is a function of the dynamic range of radiance values in the scene. Suppose the range of maximum to minimum radiance values that we are

⁵Blooming occurs when charge or light at highly saturated sites on the imaging surface spills over and affects values at neighboring sites.

⁶The working range of the film corresponds to the middle section of the response curve. The ends of the curve, in which large changes in exposure cause only small changes in density (or pixel value), are called the *toe* and the *shoulder*.

A. Anhang

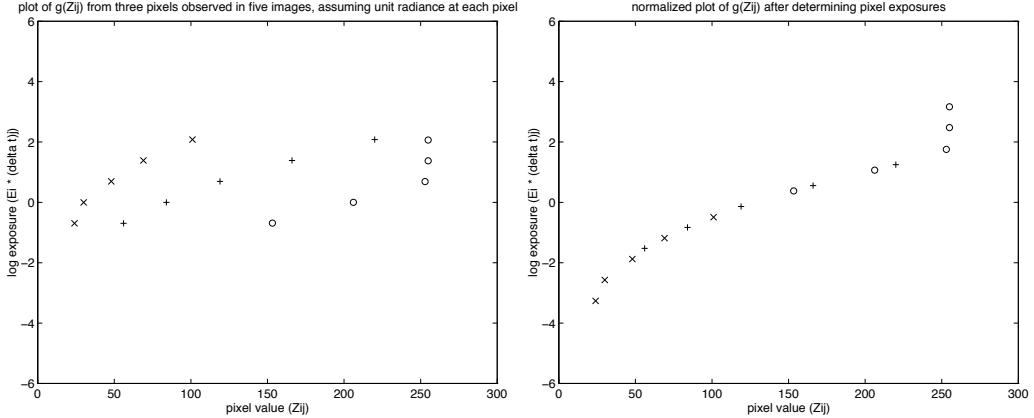


Figure 2: In the figure on the left, the \times symbols represent samples of the g curve derived from the digital values at one pixel for 5 different known exposures using Equation 2. The unknown \log irradiance $\ln E_i$ has been arbitrarily assumed to be 0. Note that the shape of the g curve is correct, though its position on the vertical scale is arbitrary corresponding to the unknown $\ln E_i$. The $+$ and \circ symbols show samples of g curve segments derived by consideration of two other pixels; again the vertical position of each segment is arbitrary. Essentially, what we want to achieve in the optimization process is to slide the 3 sampled curve segments up and down (by adjusting their $\ln E_i$'s) until they "line up" into a single smooth, monotonic curve, as shown in the right figure. The vertical position of the composite curve will remain arbitrary.

interested in recovering accurately is R , and the film is capable of representing in its working range a dynamic range of F . Then the minimum number of photographs needed is $\lceil \frac{R}{F} \rceil$ to ensure that every part of the scene is imaged in at least one photograph at an exposure duration that puts it in the working range of the film response curve. As in recovering the response curve, using more photographs than strictly necessary will result in better noise sensitivity.

If one wanted to use as few photographs as possible, one might first recover the response curve of the imaging process by photographing a scene containing a diverse range of radiance values at three or four different exposures, differing by perhaps one or two stops. This response curve could be used to determine the working range of the imaging process, which for the processes we have seen would be as many as five or six stops. For the remainder of the shoot, the photographer could decide for any particular scene the number of shots necessary to cover its entire dynamic range. For diffuse indoor scenes, only one exposure might be necessary; for scenes with high dynamic range, several would be necessary. By recording the exposure amount for each shot, the images could then be converted to radiance maps using the pre-computed response curve.

2.4 Recovering extended dynamic range from single exposures

Most commercially available film scanners can detect reasonably close to the full range of useful densities present in film. However, many of these scanners (as well as the Kodak PhotoCD process) produce 8-bit-per-channel images designed to be viewed on a screen or printed on paper. Print film, however, records a significantly greater dynamic range than can be displayed with either of these media. As a result, such scanners deliver only a portion of the detected dynamic range of print film in a single scan, discarding information in either high or low density regions. The portion of the detected dynamic range that is delivered can usually be influenced by "brightness" or "density adjustment" controls.

The method presented in this paper enables two methods for recovering the full dynamic range of print film which we will briefly

outline⁷. In the first method, the print negative is scanned with the scanner set to scan slide film. Most scanners will then record the entire detectable dynamic range of the film in the resulting image. As before, a series of differently exposed images of the same scene can be used to recover the response function of the imaging system with each of these scanner settings. This response function can then be used to convert individual exposures to radiance maps. Unfortunately, since the resulting image is still 8-bits-per-channel, this results in increased quantization.

In the second method, the film can be scanned twice with the scanner set to different density adjustment settings. A series of differently exposed images of the same scene can then be used to recover the response function of the imaging system at each of these density adjustment settings. These two response functions can then be used to combine two scans of any single negative using a similar technique as in Section 2.2.

2.5 Obtaining Absolute Radiance

For many applications, such as image processing and image compositing, the relative radiance values computed by our method are all that are necessary. If needed, an approximation to the scaling term necessary to convert to absolute radiance can be derived using the ASA of the film⁸ and the shutter speeds and exposure amounts in the photographs. With these numbers, formulas that give an approximate prediction of film response can be found in [9]. Such an approximation can be adequate for simulating visual artifacts such as glare, and predicting areas of scotopic retinal response. If desired, one could recover the scaling factor precisely by photographing a calibration luminaire of known radiance, and scaling the radiance values to agree with the known radiance of the luminaire.

2.6 Color

Color images, consisting of red, green, and blue channels, can be processed by reconstructing the imaging system curve for

⁷This work was done in collaboration with Gregory Ward Larson

⁸Conveniently, most digital cameras also specify their sensitivity in terms of ASA.

each channel independently. Unfortunately, there will be three unknown scaling factors relating relative radiance to absolute radiance, one for each channel. As a result, different choices of these scaling factors will change the color balance of the radiance map.

By default, the algorithm chooses the scaling factor such that a pixel with value Z_{mid} will have unit exposure. Thus, any pixel with the RGB value $(Z_{mid}, Z_{mid}, Z_{mid})$ will have equal radiance values for R, G, and B, meaning that the pixel is achromatic. If the three channels of the imaging system actually do respond equally to achromatic light in the neighborhood of Z_{mid} , then our procedure correctly reconstructs the relative radiances.

However, films are usually calibrated to respond achromatically to a particular color of light C , such as sunlight or fluorescent light. In this case, the radiance values of the three channels should be scaled so that the pixel value $(Z_{mid}, Z_{mid}, Z_{mid})$ maps to a radiance with the same color ratios as C . To properly model the color response of the entire imaging process rather than just the film response, the scaling terms can be adjusted by photographing a calibration luminaire of known color.

2.7 Taking virtual photographs

The recovered response functions can also be used to map radiance values back to pixel values for a given exposure Δt using Equation 1. This process can be thought of as taking a virtual photograph of the radiance map, in that the resulting image will exhibit the response qualities of the modeled imaging system. Note that the response functions used need not be the same response functions used to construct the original radiance map, which allows photographs acquired with one imaging process to be rendered as if they were acquired with another.

3 Results

Figures 3-5 show the results of using our algorithm to determine the response curve of a DCS460 digital camera. Eleven grayscale photographs filtered down to 765×509 resolution (Fig. 3) were taken at f/8 with exposure times ranging from $\frac{1}{30}$ of a second to 30 seconds, with each image receiving twice the exposure of the previous one. The film curve recovered by our algorithm from 45 pixel locations observed across the image sequence is shown in Fig. 4. Note that although CCD image arrays naturally produce linear output, from the curve it is evident that the camera nonlinearly remaps the data, presumably to mimic the response curves found in film. The underlying registered $(E_i \Delta t_j, Z_{ij})$ data are shown as light circles underneath the curve; some outliers are due to sensor artifacts (light horizontal bands across some of the darker images.)

Fig. 5 shows the reconstructed high dynamic range radiance map. To display this map, we have taken the logarithm of the radiance values and mapped the range of these values into the range of the display. In this representation, the pixels at the light regions do not saturate, and detail in the shadow regions can be made out, indicating that all of the information from the original image sequence is present in the radiance map. The large range of values present in the radiance map (over four orders of magnitude of useful dynamic range) is shown by the values at the marked pixel locations.

Figure 6 shows sixteen photographs taken inside a church with a Canon 35mm SLR camera on Fuji 100 ASA color print film. A fish-eye 15mm lens set at f/8 was used, with exposure times ranging from 30 seconds to $\frac{1}{1000}$ of a second in 1-stop increments. The film was developed professionally and scanned in using a Kodak PhotoCD film scanner. The scanner was set so that it would not individually

⁹Note that here we are assuming that the spectral response functions for each channel of the two imaging processes is the same. Also, this technique does not model many significant qualities of an imaging system such as film grain, chromatic aberration, blooming, and the modulation transfer function.



Figure 3: (a) Eleven grayscale photographs of an indoor scene acquired with a Kodak DCS460 digital camera, with shutter speeds progressing in 1-stop increments from $\frac{1}{30}$ of a second to 30 seconds.

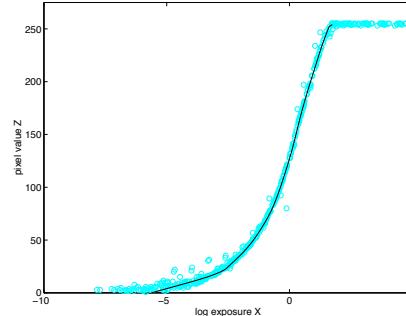


Figure 4: The response function of the DCS460 recovered by our algorithm, with the underlying $(E_i \Delta t_j, Z_{ij})$ data shown as light circles. The logarithm is base e .



Figure 5: The reconstructed high dynamic range radiance map, mapped into a grayscale image by taking the logarithm of the radiance values. The relative radiance values of the marked pixel locations, clockwise from lower left: 1.0, 46.2, 1907.1, 15116.0, and 18.0.

A. Anhang



Figure 6: Sixteen photographs of a church taken at 1-stop increments from 30 sec to $\frac{1}{1000}$ sec. The sun is directly behind the rightmost stained glass window, making it especially bright. The blue borders seen in some of the image margins are induced by the image registration process.

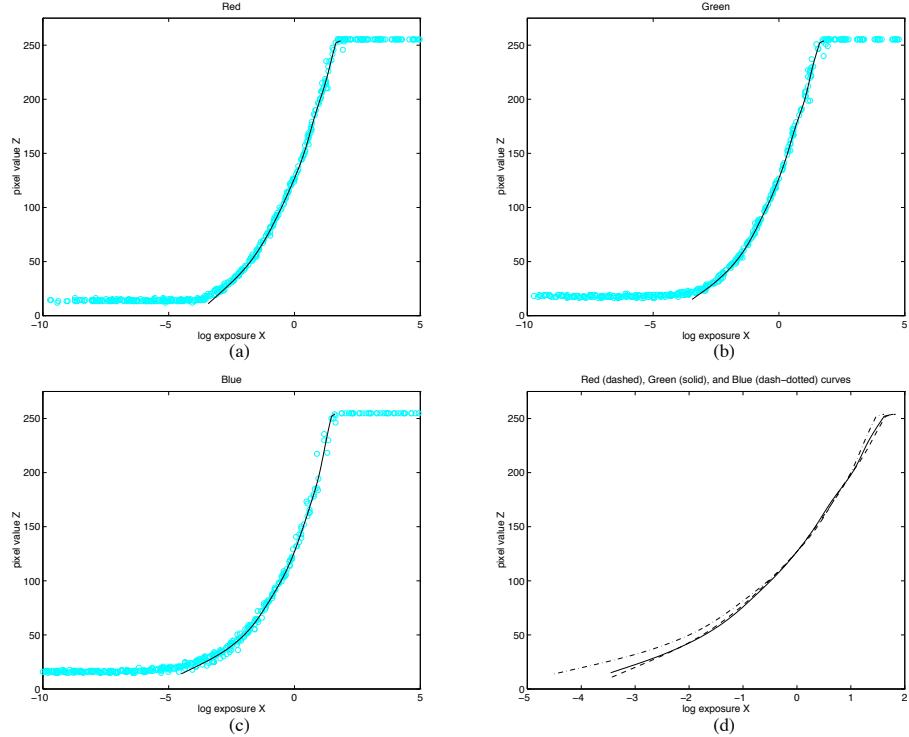


Figure 7: Recovered response curves for the imaging system used in the church photographs in Fig. 8. (a-c) Response functions for the red, green, and blue channels, plotted with the underlying $(E_i \Delta t_j, Z_{ij})$ data shown as light circles. (d) The response functions for red, green, and blue plotted on the same axes. Note that while the red and green curves are very consistent, the blue curve rises significantly above the others for low exposure values. This indicates that dark regions in the images exhibit a slight blue cast. Since this artifact is recovered by the response curves, it does not affect the relative radiance values.

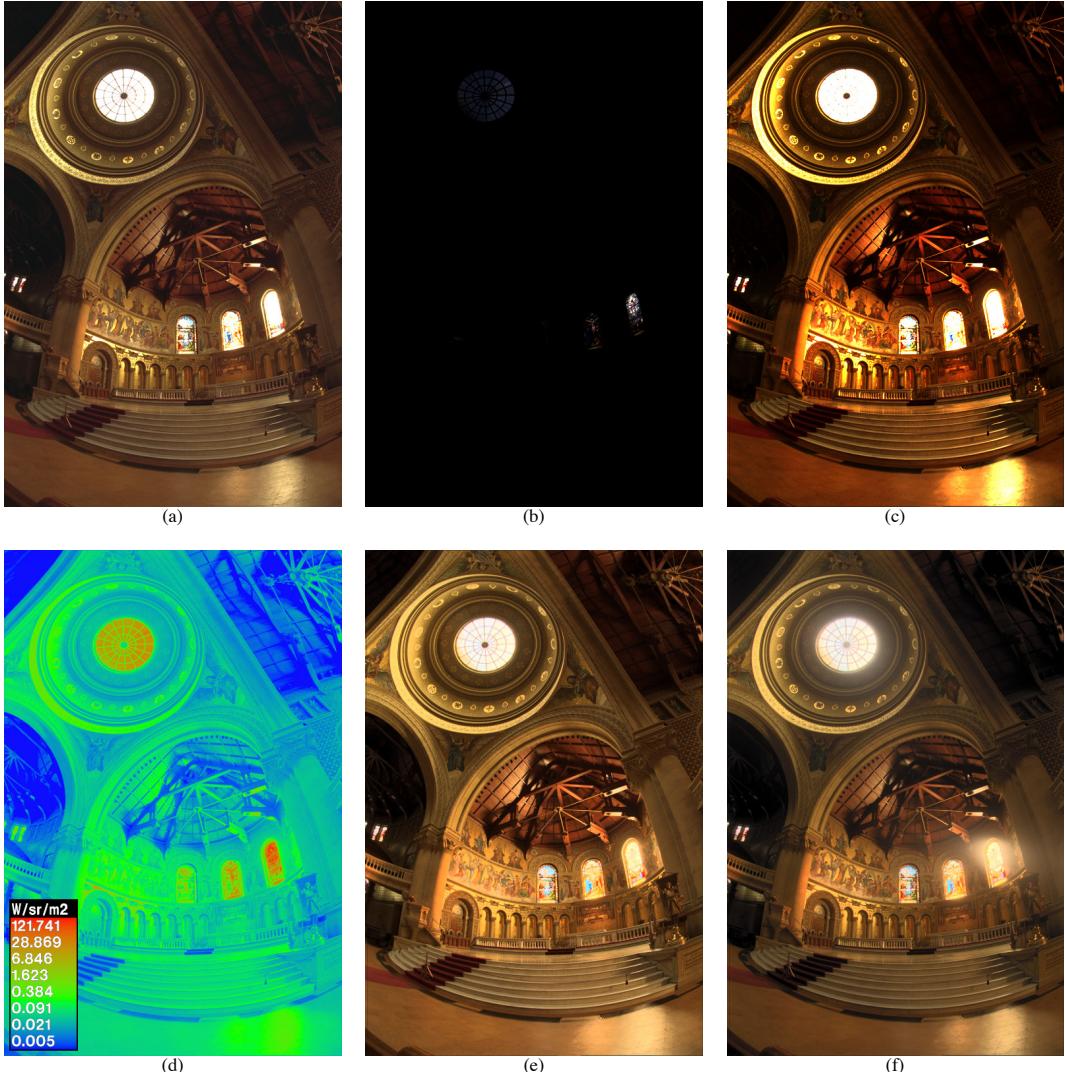


Figure 8: (a) An actual photograph, taken with conventional print film at two seconds and scanned to PhotoCD. (b) The high dynamic range radiance map, displayed by linearly mapping its entire dynamic range into the dynamic range of the display device. (c) The radiance map, displayed by linearly mapping the lower 0.1% of its dynamic range to the display device. (d) A false-color image showing relative radiance values for a grayscale version of the radiance map, indicating that the map contains over five orders of magnitude of useful dynamic range. (e) A rendering of the radiance map using adaptive histogram compression. (f) A rendering of the radiance map using histogram compression and also simulating various properties of the human visual system, such as glare, contrast sensitivity, and scotopic retinal response. Images (e) and (f) were generated by a method described in [23]. Images (d-f) courtesy of Gregory Ward Larson.

A. Anhang

adjust the brightness and contrast of the images¹⁰ to guarantee that each image would be digitized using the same response function.

An unfortunate aspect of the PhotoCD process is that it does not scan precisely the same area of each negative relative to the extents of the image.¹¹ To counteract this effect, we geometrically registered the images to each other using a using normalized correlation (see [4]) to determine, with sub-pixel accuracy, corresponding pixels between pairs of images.

Fig. 7(a-c) shows the response functions for the red, green, and blue channels of the church sequence recovered from 28 pixel locations. Fig. 7(d) shows the recovered red, green, and blue response curves plotted on the same set of axes. From this plot, we can see that while the red and green curves are very consistent, the blue curve rises significantly above the others for low exposure values. This indicates that dark regions in the images exhibit a slight blue cast. Since this artifact is modeled by the response curves, it will not affect the relative radiance values.

Fig. 8 interprets the recovered high dynamic range radiance map in a variety of ways. Fig. 8(a) is one of the actual photographs, which lacks detail in its darker regions at the same time that many values within the two rightmost stained glass windows are saturated. Figs. 8(b,c) show the radiance map, linearly scaled to the display device using two different scaling factors. Although one scaling factor is one thousand times the other, there is useful detail in both images. Fig. 8(d) is a false-color image showing radiance values for a grayscale version of the radiance map; the highest listed radiance value is nearly 250,000 times that of the lowest. Figs. 8(e,f) show two renderings of the radiance map using a new tone reproduction algorithm [23]. Although the rightmost stained glass window has radiance values over a thousand times higher than the darker areas in the rafters, these renderings exhibit detail in both areas.

Figure 9 demonstrates two applications of the techniques presented in this paper: accurate signal processing and virtual photography. The task is to simulate the effects of motion blur caused by moving the camera during the exposure. Fig. 9(a) shows the results of convolving an actual, low-dynamic range photograph with a 37×1 pixel box filter to simulate horizontal motion blur. Fig. 9(b) shows the results of applying this same filter to the high dynamic range radiance map, and then sending this filtered radiance map back through the recovered film response functions using the same exposure time Δt as in the actual photograph. Because we are seeing this image through the actual image response curves, the two left images are tonally consistent with each other. However, there is a large difference between these two images near the bright spots. In the photograph, the bright radiance values have been clamped to the maximum pixel values by the response function. As a result, these clamped values blur with lower neighboring values and fail to saturate the image in the final result, giving a muddy appearance.

In Fig. 9(b), the extremely high pixel values were represented properly in the radiance map and thus remained at values above the level of the response function's saturation point within most of the blurred region. As a result, the resulting virtual photograph exhibits several crisply-defined saturated regions.

Fig. 9(c) is an actual photograph with real motion blur induced by spinning the camera on the tripod during the exposure, which is equal in duration to Fig. 9(a) and the exposure simulated in Fig. 9(b). Clearly, in the bright regions, the blurring effect is qualitatively similar to the synthetic blur in 9(b) but not 9(a). The precise shape of the real motion blur is curved and was not modeled for this demonstration.

¹⁰This feature of the PhotoCD process is called “Scene Balance Adjustment”, or SBA.

¹¹This is far less of a problem for cinematic applications, in which the film sprocket holes are used to expose and scan precisely the same area of each frame.



(a) Synthetically blurred digital image



(b) Synthetically blurred radiance map



(c) Actual blurred photograph

Figure 9: (a) Synthetic motion blur applied to one of the original digitized photographs. The bright values in the windows are clamped before the processing, producing mostly unsaturated values in the blurred regions. (b) Synthetic motion blur applied to a recovered high-dynamic range radiance map, then virtually photographed through the recovered film response curves. The radiance values are clamped to the display device after the processing, allowing pixels to remain saturated in the window regions. (c) Real motion blur created by rotating the camera on the tripod during the exposure, which is much more consistent with (b) than (a).

4 Conclusion

We have presented a simple, practical, robust and accurate method of recovering high dynamic range radiance maps from ordinary photographs. Our method uses the constraint of sensor reciprocity to derive the response function and relative radiance values directly from a set of images taken with different exposures. This work has a wide variety of applications in the areas of image-based modeling and rendering, image processing, and image compositing, a few of which we have demonstrated. It is our hope that this work will be able to help both researchers and practitioners of computer graphics make much more effective use of digitized photographs.

Acknowledgments

The authors wish to thank Tim Hawkins, Carlo Séquin, David Forsyth, Steve Chenney, Chris Healey, and our reviewers for their valuable help in revising this paper. This research was supported by a Multidisciplinary University Research Initiative on three dimensional direct visualization from ONR and BMDO, grant FDN00014-96-1-1200.

References

- [1] ADAMS, A. *Basic Photo*, 1st ed. Morgan & Morgan, Hastings-on-Hudson, New York, 1970.
- [2] CHEN, E. QuickTime VR - an image-based approach to virtual environment navigation. In *SIGGRAPH '95* (1995).
- [3] DEBEVEC, P. E., TAYLOR, C. J., AND MALIK, J. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *SIGGRAPH '96* (August 1996), pp. 11–20.
- [4] FAUGERAS, O. *Three-Dimensional Computer Vision*. MIT Press, 1993.
- [5] FERWERDA, J. A., PATTANAIK, S. N., SHIRLEY, P., AND GREENBERG, D. P. A model of visual adaptation for realistic image synthesis. In *SIGGRAPH '96* (1996), pp. 249–258.
- [6] GORTLER, S. J., GRZESZCZUK, R., SZELISKI, R., AND COHEN, M. F. The Lumigraph. In *SIGGRAPH '96* (1996), pp. 43–54.
- [7] HORN, B. K. P. *Robot Vision*. MIT Press, Cambridge, Mass., 1986, ch. 10, pp. 206–208.
- [8] JAMES, T., Ed. *The Theory of the Photographic Process*. Macmillan, New York, 1977.
- [9] KAUFMAN, J. E., Ed. *IES Lighting Handbook; the standard lighting guide*, 7th ed. Illuminating Engineering Society, New York, 1987, p. 24.
- [10] KOLB, C., MITCHELL, D., AND HANRAHAN, P. A realistic camera model for computer graphics. In *SIGGRAPH '95* (1995).
- [11] LAVEAU, S., AND FAUGERAS, O. 3-D scene representation as a collection of images. In *Proceedings of 12th International Conference on Pattern Recognition* (1994), vol. 1, pp. 689–691.
- [12] LEVOY, M., AND HANRAHAN, P. Light field rendering. In *SIGGRAPH '96* (1996), pp. 31–42.
- [13] MADDEN, B. C. Extended intensity range imaging. Tech. rep., GRASP Laboratory, University of Pennsylvania, 1993.
- [14] MANN, S., AND PICARD, R. W. Being ‘undigital’ with digital cameras: Extending dynamic range by combining differently exposed pictures. In *Proceedings of IS&T 46th annual conference* (May 1995), pp. 422–428.
- [15] MCMILLAN, L., AND BISHOP, G. Plenoptic Modeling: An image-based rendering system. In *SIGGRAPH '95* (1995).
- [16] SCHLICK, C. Quantization techniques for visualization of high dynamic range pictures. In *Fifth Eurographics Workshop on Rendering* (Darmstadt, Germany) (June 1994), pp. 7–18.
- [17] SZELISKI, R. Image mosaicing for tele-reality applications. In *IEEE Computer Graphics and Applications* (1996).
- [18] TANI, T. *Photographic sensitivity: theory and mechanisms*. Oxford University Press, New York, 1995.
- [19] THEUWISSEN, A. J. P. *Solid-state imaging with charge-coupled devices*. Kluwer Academic Publishers, Dordrecht; Boston, 1995.
- [20] TUMBLIN, J., AND RUSHMEIER, H. Tone reproduction for realistic images. *IEEE Computer Graphics and Applications* 13, 6 (1993), 42–48.
- [21] WARD, G. J. Measuring and modeling anisotropic reflection. In *SIGGRAPH '92* (July 1992), pp. 265–272.
- [22] WARD, G. J. The radiance lighting simulation and rendering system. In *SIGGRAPH '94* (July 1994), pp. 459–472.
- [23] WARD, G. J., RUSHMEIER, H., AND PIATKO, C. A visibility matching tone reproduction operator for high dynamic range scenes. Tech. Rep. LBNL-39882, Lawrence Berkeley National Laboratory, March 1997.

A Matlab Code

Here is the MATLAB code used to solve the linear system that minimizes the objective function \mathcal{O} in Equation 3. Given a set of observed pixel values in a set of images with known exposures, this routine reconstructs the imaging response curve and the radiance values for the given pixels. The weighting function $w(z)$ is found in Equation 4.

```
% gsolve.m - Solve for imaging system response function
%
% Given a set of pixel values observed for several pixels in several
% images at different exposure times, this function returns the
% imaging system's response function g as well as the log film irradiance
% values for the observed pixels.
%
% Assumes:
%
% Zmin = 0
% Zmax = 255
%
% Arguments:
%
% Z(i,j) is the pixel values of pixel location number i in image j
% B(j) is the log delta t, or log shutter speed, for image j
% l is lambda, the constant that determines the amount of smoothness
% w(z) is the weighting function value for pixel value z
%
% Returns:
%
% g(z) is the log exposure corresponding to pixel value z
% IE(i) is the log film irradiance at pixel location i
%
function [g,IE]=gsolve(Z,B,l,w)
n = 256;
A = zeros(size(Z,1)*size(Z,2)+n+1,n+size(Z,1));
b = zeros(size(A,1),1);
%
% Include the data-fitting equations
k = 1;
for i=1:size(Z,1)
    for j=1:size(Z,2)
        wij = w(Z(i,j)+1);
        A(k,i+j) = wij; A(k,n+i) = -wij; b(k,1) = wij * B(i,j);
        k=k+1;
    end
%
% Fix the curve by setting its middle value to 0
A(k,129) = 1;
k=k+1;
%
% Include the smoothness equations
for i=1:n-2
    A(k,i+1) = 1*w(i+1); A(k,i+2) = -2*w(i+1); A(k,i+3) = 1*w(i+1);
    k=k+1;
end
%
% Solve the system using SVD
x = A\b;
g = x(1:n);
IE = x(n+1:size(x,1));
```

A.2. Verwendete Algorithmen

A.2.1. 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 Bilderpyramide 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)
```

Listing A.2 Rekursive C Funktion zur Berechnung der notwendigen Verschiebung zwischen den Bildern um diese zu registrieren [Waro3]

```
/** 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.2.2. 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

—. 15

CMOS

Complementary Metal Oxide Semiconductor. 17, 26

Dynamikumfang

Verhältnis von größter und kleinster Leuchtdichte.. 15, 16, 20

GUI

grafische Benutzerschnittstelle (engl. graphical user interface). 49, 50

HDR

High Dynamic Range. 7, 11, 13, 15–22, 24–28, 33

JRE

Java Runtime Environment. 50

LDR

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

LGS

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

Monotonie

Test test. 12, 61

MTB

Mean Threshold Bitmap Alignment. 19, 80

MVC

Model-View-Controller. 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. 15, 20, 29, 30, 33, 52, 62

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, 12, 42, 43, 59, 62, 64

SOR

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

SVD

singular value decomposition (dt. Singulärwertzerlegung). 29

Tone-Mapping

Test test. 7, 8, 13, 16, 20–22, 24, 26, 51, 54, 59, 60

Literaturverzeichnis

- [Ado92] Adobe. Tiff specification, Revision 6.0, 1992. URL <http://partners.adobe.com/public/developer/en/tiff/TIFF6.pdf>. (Zitiert auf Seite 20)
- [BGH⁺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 18)
- [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, 20 und 21)
- [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 18)
- [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, 6, 7, 11, 12, 13, 23, 27, 28, 30, 71, 73, 75, 77, 79 und 81)
- [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 15)
- [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⁺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 Seite 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. Licher. *Software Engineering*. dpunkt.verlag GmbH, Heidelberg, 2 Auflage, 2010. (Zitiert auf den Seiten 51 und 52)
- [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 24)
- [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 Seite 21)
- [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, 15, 16, 17, 19, 20 und 82)
- [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, 16, 17 und 24)
- [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, 82 und 83)
- [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 Seite 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