Mollifying Realistic Image Synthesis for Time Constrained Rendering

Zur Erlangung des akademischen Grades einer

Doktorin der Ingenieurwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Alisa Jung

aus Mühlacker

Tag der mündlichen Prüfung: 16. Mai 2024

1. Referent: Prof. Dr.-Ing. Carsten Dachsbacher

2. Referent: Prof. Dr. techn. Alexander Wilkie

Acknowledgements

I'd like to thank the many people who supported me during my doctorate, for their advice, guidance, feedback, and encouragement:

First of all, I'm extremely grateful to my supervisor, Carsten Dachsbacher, whose guidance and critical questions helped me stay on track with my projects and encouraged me to dive deeper into problems, for giving me the freedom and confidence to explore different topics, and for making this thesis possible in the first place.

I would also like to thank everyone at the computer graphics group at KIT, with Diana who always looks out for us, Johannes, for always being there to talk through problems, Vincent, for being an amazing office mate, Florian, who got me excited for graphics early in my studies, Christoph, for showing me the importance of proper maths, Addis, Colin and Killian, who took the time to do some thesis proof-reading, and of course everyone on the team for always being there to talk through ideas or roadblocks, cheering each other up, and the good times we had along the way at Tischkicker, puzzles and cucinare bene, Boris, Christoph, Emanuel, Hisanari, Johannes, Lorenzo, Mahmoud, Max, Mikhail, Moritz, Reiner, Tobias, and Tobias.

I very much appreciate my co-authors Alexander, Wenzel, Steve, Eric, and Lars, who gave me so much inspiration for new ideas and taught me a lot about paper writing, with a special thanks to Alexander for also taking the time to be second reviewer for this thesis.

I also had the great pleasure of working with the fantastic folks at Wētā, Marc who was always open to new ideas while making sure we kept going in the right direction, Jirka for his endless patience discussing guiding peculiarities, Barton, Carla, Chris, Courtney, Guillaume, Javor, Julia, Kerry, Lucia, Manuele, Max, Nick, Remi, and Sander.

And of course special thanks to my family and friends, for their infinite emotional support and always believing in me, in particular Yan and Isabell, Joanna, Morten, Jonas, Jannik, Barbara and Petra, Micha and Johannes, Anna and Gem, and my ultimate frisbee teams Colorado, Candy and Nexus.

Without you wonderful people this thesis would not have been possible. Thank you!

Contents

1 Zusammenfassung **1**
- 1.1 Einführung . . . 1
- 1.2 Beiträge . . . 2

2 Summary **7**
- 2.1 Introduction . . . 7
- 2.2 Contributions . . . 8

3 List of publications **13**

4 Background **15**
- 4.1 Measures and Integration . . . 15
  - 4.1.1 Definitions . . . 15
  - 4.1.2 Changing the Integration Domain . . . 17
  - 4.1.3 Examples . . . 18
- 4.2 Probability Theory . . . 21
  - 4.2.1 Random Variables . . . 21
  - 4.2.2 Expected Value and Variance . . . 25
  - 4.2.3 Random Number Generation . . . 26
- 4.3 Monte Carlo Integration . . . 28
  - 4.3.1 Numerical Integration, Bias and Consistency . . . 28
  - 4.3.2 The Monte Carlo Estimator . . . 29
  - 4.3.3 Importance Sampling . . . 31
  - 4.3.4 Multiple Importance Sampling . . . 31
- 4.4 Photorealistic Image Synthesis: Theory . . . 33
  - 4.4.1 Radiometry . . . 33
  - 4.4.2 Light Sources . . . 36
  - 4.4.3 BSDF . . . 38
  - 4.4.4 Participating Media . . . 42
  - 4.4.5 Pixel Sensors and Camera Models . . . 42
  - 4.4.6 Rendering Equation . . . 43
  - 4.4.7 Path Space . . . 44
  - 4.4.8 Path Classification . . . 46
- 4.5 Photorealistic Image Synthesis: Solutions . . . 47
  - 4.5.1 Forward Path Tracing . . . 47
  - 4.5.2 Next Event Estimation . . . 48
  - 4.5.3 Light Tracing . . . 50

4.5.4 Bidirectional Path Tracing . . . 51
4.5.5 Discussion . . . 52
4.6 Spectral Light Transport . . . 54
4.6.1 Color Spaces . . . 55
4.6.2 Problems with RGB Rendering . . . 56
4.6.3 Spectral Rendering . . . 58
4.6.4 Spectral Upsampling . . . 59
4.7 Fluorescence . . . 61
4.7.1 Background . . . 61
4.7.2 Reradiation at Shorter Wavelengths . . . 65
4.7.3 Fluorescent BBRRDF . . . 68
4.7.4 Extended Rendering Equation . . . 71
4.7.5 Rendering Fluorescence . . . 72
4.8 Singularities in the Rendering Equation . . . 75
4.8.1 Singularity Models . . . 76
4.8.2 Distributions . . . 80
4.8.3 Mollification . . . 82
4.9 Evaluation of Renderer Implementations . . . 85
4.9.1 Rendering Artifacts . . . 85
4.9.2 Color Difference Metrics . . . 85
4.9.3 Image Metrics . . . 86

5 Wide Gamut Spectral Upsampling with Fluorescence 89
5.1 Introduction . . . 90
5.2 Background and Previous Work . . . 91
5.3 Fluorescent RGB Upsampling . . . 93
5.3.1 Overview . . . 93
5.3.2 Elastic Scattering Component . . . 93
5.3.3 A Parametric Model for Fluorescent Spectra . . . 94
5.3.4 Parameter Search . . . 97
5.3.5 Importance Sampling . . . 99
5.4 Results . . . 100
5.5 Extensions . . . 105
5.5.1 Perception Based Color Distance Metric . . . 106
5.5.2 Fluorescent Dyes . . . 106
5.5.3 Alternative Interpolation . . . 107
5.5.4 Results . . . 108
5.6 Conclusion and Future Work . . . 110

6 Spectral Mollification for Bidirectional Fluorescence 113
6.1 Introduction . . . 113
6.2 Background . . . 117
6.2.1 Spectral Singularities . . . 117
6.3 Spectral Mollification for Bidirectional Fluorescence . . . 120
6.3.1 Mollifying Reradiation Matrices . . . 120
6.3.2 Mollifying a Non-Fluorescent BBRRDF . . . 120
6.3.3 Mollified PDF . . . 122
6.3.4 Sampling Strategies . . . 123

6.3.5 Multiple Importance Sampling Weights . . . 123
6.3.6 Consistency of the Estimator . . . 124
6.4 Evaluation . . . 125
6.4.1 Scene Setups . . . 125
6.4.2 Color Bias . . . 127
6.4.3 Constant Mollification vs. Box Filtering . . . 129
6.4.4 Bidirectional vs. Unidirectional . . . 130
6.4.5 Mollified vs. Unmollified BDPT . . . 132
6.4.6 Convergence Rate . . . 134
6.5 Discussion . . . 137
6.6 Limitations and Future Work . . . 137
6.7 Conclusion . . . 138
6.8 Acknowledgments . . . 138

7 Spatial Mollification for Next Event Estimation with Small Lights 139
7.1 Introduction . . . 139
7.2 Background . . . 141
7.2.1 Regularization . . . 141
7.2.2 Vertex Connection and Merging . . . 142
7.3 Related Work . . . 142
7.3.1 Regularization for Monte Carlo Rendering . . . 142
7.3.2 Regularization Kernels . . . 143
7.3.3 Specular Next Event Estimation without Regularization . . . 144
7.3.4 MIS with Biased Estimators . . . 144
7.4 Overview . . . 144
7.5 Merge-Based NEE Regularization . . . 146
7.5.1 NEE Merging without Singularities . . . 146
7.5.2 NEE Merging with One Singularity . . . 148
7.5.3 NEE Merging with Two Singularities . . . 149
7.5.4 Forcing the BSDF Hemisphere . . . 151
7.6 Adaptive Regularization Kernels . . . 151
7.6.1 Kernel Shapes for Point or Area Lights . . . 152
7.6.2 Kernel Scale . . . 152
7.7 Multiple Importance Sampling: VCM-Style . . . 153
7.8 Variance-Minimizing MIS Weights . . . 155
7.8.1 Individual Variance Analysis . . . 155
7.8.2 Area-based Weights . . . 157
7.8.3 Combined Variance . . . 157
7.8.4 Limit Analysis . . . 158
7.8.5 Approximating the Overlap . . . 158
7.8.6 Forcing the BSDF Hemisphere . . . 159
7.8.7 Multiple Emitters . . . 159
7.9 Results . . . 160
7.9.1 Kernel Scales . . . 160
7.9.2 Kernel Shapes . . . 161
7.9.3 Visibility . . . 163
7.9.4 Multiple Light Sources . . . 163
7.9.5 Environment Maps . . . 165

7.9.6 MIS . . . 165
7.9.7 Consistency . . . 167
7.10 Limitations and Future Work . . . 168
7.11 Conclusion . . . 169

8 Detecting Bias in Monte Carlo Renderers using Welch's t-test 171
8.1 Introduction . . . 172
8.2 Related Work . . . 172
8.3 Welch's t-Test . . . 173
8.3.1 Testing for Equal Means . . . 174
8.3.2 Properties of the p-Value . . . 175
8.3.3 Gaussian Approximation . . . 176
8.4 Application to Monte Carlo Rendering . . . 177
8.4.1 Generating Normal Distributed Samples . . . 177
8.4.2 Sample Preparation during Rendering . . . 178
8.4.3 Considerations for Halton Points . . . 179
8.4.4 Visualization . . . 179
8.5 Results . . . 181
8.5.1 Welch Sample Generation . . . 181
8.5.2 Unbiased Reference Test . . . 183
8.5.3 Finding Bias in a Bidirectional Path Tracer . . . 183
8.5.4 Detecting a Biased Scene . . . 184
8.5.5 Biased MIS Weights in a Path Tracer . . . 184
8.5.6 Influence of the Pixel Tile Size . . . 185
8.5.7 Gaussian Approximation . . . 186
8.5.8 Halton Points . . . 188
8.5.9 Run Time and Memory Consumption . . . 188
8.6 Conclusion . . . 189

9 Conclusion 191

A Appendix: Background 193
A.1 Light Transport as a Measure . . . 193

B Appendix: Spectral Mollification for Bidirectional Fluorescence 195
B.1 Energy Conservation . . . 195
B.2 Photon Conservation . . . 196
B.2.1 Symmetric Kernels . . . 196
B.2.2 Asymmetric Kernels . . . 198
B.3 Mollified PDFs . . . 198
B.3.1 Constant Mollification . . . 198
B.3.2 Box Mollification . . . 200
B.3.3 Reradiation Matrices . . . 200
B.4 Additional Results . . . 201

C Appendix: Spatial Mollification for Next Event Estimation with Small Lights **203**
C.1 Specular and Refractive Chains 203
C.1.1 Where the Jacobian comes from 204
C.1.2 Pitfalls with the Dirac Delta Notation 206
C.2 Environment Maps 207
C.3 Minimizing the Variance of the Combined Estimate 208
C.3.1 Specular Chains and Refractions 209
C.4 Analytical Overlap of Simple Light and Kernel Shapes 210
C.5 Additional Results 211
C.5.1 Shape Overlap 211
C.5.2 MIS Weights 213

Bibliography **215**

1 Zusammenfassung

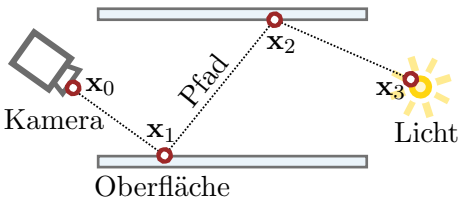
Die fotorealistische Bildsynthese nutzt eine physikalisch-basierte Lichttransportsimulation, um aus einer virtuellen Beschreibung einer Szene Bilder zu erzeugen (*Rendering*). Eingesetzt wird sie etwa in der Produkt- und Architekturvisualisierung, zur prädiktiven Bildsynthese (z.B. bei der Vorschau neuer Materialien, Lichtsimulationen für Solarzellen, Generierung von Trainingsdaten im Kontext von maschinellem Lernen, oder um die energetischen Eigenschaften von Gebäuden zu simulieren), sowie in der Filmindustrie. Hierfür wird der Strahlungstransport, also die Ausbreitung des Lichts in einer virtuellen Szene, mit numerischen Methoden simuliert. Die Szene bildet dabei die Simulationsparameter.

Diese Arbeit beschäftigt sich mit einigen Schritten auf dem Weg dieser Simulation. Zunächst wird in Kapitel 5 untersucht, wie als RGB-Tripel vorhandene Eingabedaten in wellenlängenabhängige Spektraldaten umgewandelt werden können, sodass die Lichttransportsimulation die Wellenlänge des Lichts berücksichtigen kann. Daraufhin stellen Kapitel 6 und 7 zwei Verfahren vor, die in besonders problematischen Konfigurationen sowohl Konvergenzrate als auch Rauschverhalten stark verbessern können, indem ein leichter Fehler in das zugrundeliegende Integrationsproblem eingeführt wird. Zum Abschluss stellt Kapitel 8 ein neues Verfahren vor, welches systematische Fehler in der Simulation bereits nach kurzer Rechenzeit detektieren kann.

1.1 Einführung

Das Ziel der fotorealistischen Bildsynthese ist, das auf einem virtuellen Kamerasensor einfallende Licht näherungsweise zu berechnen.
Dafür beschreibt die *Render-Gleichung*

$$I_j = \int_{\mathcal{P}} f(\mathbf{X}) \, \mathrm{d}\mathbf{X}$$



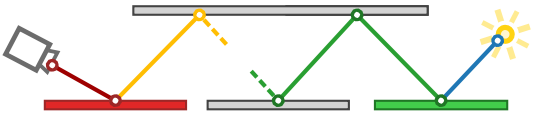
das einfallende Licht I_j an einem Kamerapixel j als Integral über den Beitrag f aller Lichttransportpfade $\mathbf{X} = (\mathbf{x}_0, \ldots, \mathbf{x}_k)$ aus dem Raum $\mathcal{P}$ aller möglichen Pfade, auf denen Photonen von der Lichtquelle zum Kamerasensor gelangen können. Der Beitrag f enthält unter anderem Terme für die Intensität der Lichtquelle und das Reflektanzverhalten der Oberflächenpunkte $\mathbf{x}_i$, und hängt von der Wellenlänge λ des Lichts ab.

Da das Integral für I_j hochdimensional ist, löst man es in der Regel aufgrund der Konvergenzeigenschaften numerisch durch *Monte Carlo-Integration*. Hierfür werden N

Lichtpfade zufällig und unabhängig voneinander stochastisch erzeugt und deren Beiträge akkumuliert. Die Varianz dieser Verfahren äußert sich als Bildrauschen und verschwindet mit steigender Stichprobengröße N.

Eine einfache Strategie, Pfade zu erzeugen, ist Path Tracing, welches einen Pfad auf einem Sensor-Pixel j startet und dann Knoten für Knoten in zufällige Richtungen durch die Szene verfolgt, bis er die Lichtquelle trifft. Als robustere Variante davon startet Bidirektionales Path Tracing ($BDPT$) je einen Teilpfad auf dem Pixel und auf der Lichtquelle, und verbindet diese.

Unabhängig von der gewählten Strategie bereiten Singularitäten in den Integranden Schwierigkeiten, da diese schlecht bis gar nicht abtastbar sind und für ein rauschfreies Ergebnis sehr viele Pfade erfordern. Eine Lösung dafür bietet *Mollifizierung*. Dabei wird eine regularisierte Variante von I_j gelöst, was bei geschickter Regularisierung in kürzerer Zeit zu einer geringeren Varianz mit einem geringen bis vernachlässigbaren systematischem Fehler führen kann.



Eine besondere Herausforderung stellen Interaktionen von Licht mit fluoreszenten Materialien dar. Diese können die Wellenlänge des einfallenden Lichts ändern. Dadurch wirkt die Farbe mancher Materialien besonders gesättigt, wie etwa Warnwesten oder Textmarker. In der Abbildung fluoreszieren die rote und grüne Oberfläche und ändern so die Wellenlänge (farbig dargestellt) entlang des Kamera- und Lichtteilpfades.

1.2 Beiträge

Die Beiträge meiner Arbeit liegen im Bereich von effizienten und robusten Monte Carlo-Bildsynthese-Verfahren mit Fokus auf neuen Ansätzen zur Mollifizierung und fluoreszenten Materialien.

Die Arbeit [1] (Kapitel 6) regularisiert in der Wellenlängendomäne und ermöglicht so Verbindungen von Kamera- und Lichtteilpfaden, die aufgrund von Fluoreszenz in unterschiedlichen Wellenlängen enden. Die Arbeit [2] (Kapitel 7) hingegen regularisiert in der geometrischen Domäne, um Verbindungen zwischen singulären Oberflächen (z.B. Spiegel) und Emittern (z.B. Punktlichter) zu ermöglichen.

Die Arbeiten [3] (Kapitel 5) und [4] (Kapitel 5.5) repräsentieren als RGB-Tripel gegebene Farbeindrücke durch Kombination von Reflektanz- und Fluoreszenzspektren, um besonders intensive Farben in einem physikalisch plausiblen Rahmen zu erhalten.

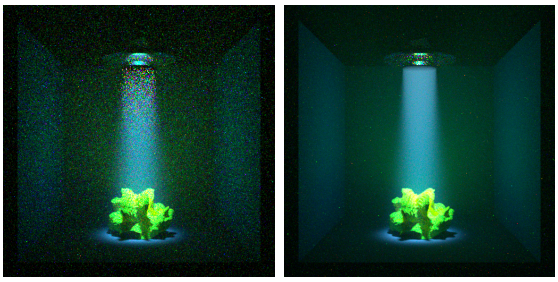
Die Entwicklung neuer Rendering-Algorithmen kann mit Fehlern verbunden sein, die kaum oder nur mit einer hohen Stichprobengröße erkennbar sind. Die Arbeit [5] (Kapitel 8) stellt ein Verfahren vor, das bereits bei kleinen Stichprobengrößen Hinweise auf Unstimmigkeiten in der Theorie und Implementierung neuer Algorithmen geben kann.

Spektrale Mollifizierung für fluoreszentes Bidirektionales Path Tracing

Üblicherweise werden bei Bidirektionalem Path Tracing je ein Lichttransportpfad von der Kamera und der Lichtquelle aus mit der selben zufälligen Wellenlänge gestartet, und dann unabhängig voneinander durch die Szene verfolgt. Man versucht dann, durch Verbindung der Teilpfade valide Gesamtpfade zu erzeugen.

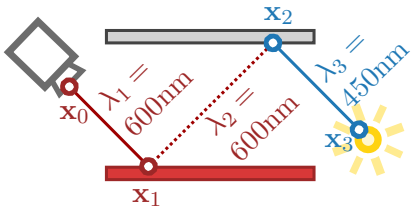
Nun können fluoreszente Oberflächen die Wellenlänge des Lichts entlang der Pfade ändern (siehe Abbildung auf Seite 2), ihre Reflektanzfunktionen $f_f(\lambda_i, \lambda_o)$ sind über Änderungen der ein- (λ_i) und ausgehenden (λ_o) Wellenlänge kontinuierlich. Klassische Reflektanzmodelle sind hingegen nur definiert, falls Licht eine Oberfläche mit der selben Wellenlänge verlässt, mit der es dort eingefallen ist. Verbindet man also einen Kamera- mit einem Lichtteilpfad, wobei die Teilpfade aufgrund von Fluoreszenz in unterscheidlichen Wellenlängen enden (siehe Abbildung auf Seite 2), so ist der Beitrag f des Ergebnispfades nicht mehr wohldefiniert.

Die Arbeit [1] mollifiziert die nicht-fluoreszente Interaktion am Verbindungsknoten, also ersetzt sie durch eine Familie leicht integrierbarer fluoreszenter Reflektanzen, die mit $N \to \infty$ simulierten Pfaden zur ursprünglichen Reflektanzfunktion konvergieren. Bei begrenzter Rechenzeit verursacht der resultierende systematische Fehler verfälschte (Farbton) oder verdunkelte (Helligkeit) Farben. Mit geschickten Mollifizierungsparametern sind diese Fehler mit dem bloßen Auge kaum bis nicht erkennbar. Bei entsprechender Anpassung der Parameter über Iterationen ist die Methode zudem konsistent, konvergiert also mit $N \to \infty$ Pfaden zum korrekten Ergebnis, während der Fehler verschwindet.



Path Tracer

Mollifizierter BDPT



Verbindung Kamerapfad $(\mathbf{x}_0, \mathbf{x}_1)$ mit roter zu Lichtpfad $(\mathbf{x}_3, \mathbf{x}_2)$ mit blauer Wellenlänge, Mollifizierung bei $\mathbf{x}_2$.

Geometrische Mollifizierung für singuläre direkte Beleuchtung

Singularitäten im Render-Integrand sind problematisch für viele Monte Carlo Path Tracing Varianten, da eine kontinuierlich erzeugte Zufallszahl keine diskrete Singularität treffen kann. Ebenso können Beinahe-Singularitäten, also Integranden mit sehr kleinem Träger, nur schwierig abzutasten sein, falls der Träger zufällig von einer anderweitig erzeugten Zufallszahl getroffen werden muss. Ein Beispiel dafür wären Punktlichtquellen bzw. kleine Flächenlichtquellen, die bei gewöhnlichem Path Tracing gar nicht bzw. nur selten von in der Szene zufällig gewählten Richtungen getroffen werden können. Next Event Estimation, also die gezielte Verbindung von Szenenoberflächen zu (beinahe)

singulären Lichtquellen, behebt dieses Problem, scheitert aber an (beinahe) singulären Oberflächen wie Spiegeln oder Glas. Treten singuläre Oberflächen und Lichtquellen gemeinsam auf, scheitern beide diese Ansätze.

In Kapitel 7 untersuchen wir verschiedene Arten, solche Singularitäten darzustellen, und leiten eine neue Regularisierungstechnik für diese Fälle ab. Zudem diskutieren wir den Zusammenhang mit Kerneldichteschätzungen und wie sich das Verhältnis von Kernelgröße und Träger einer (beinahe-)Singularität auf Varianz und systematischen Fehler auswirkt.

RGB-zu-Spektrum Rekonstruktion unter Berücksichtigung von Fluoreszenz

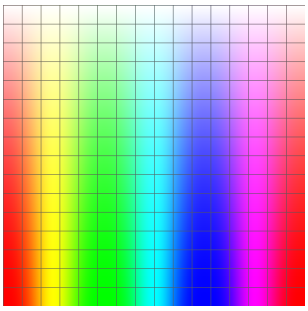
In der Praxis ignorieren viele Lichttransportsimulationen neben Fluoreszenz auch alle weiteren spektralen Effekte, und werten den Beitrag der Pfade nach deren Erzeugung separat je für einen Rot-, Grün- und Blaukanal aus. Zur Szenenmodellierung erstellte Materialien liegen daher häufig in RGB- oder anderen Tristimulusfarbräumen vor, welche Farben durch drei diskrete Werte anstatt kontinuierlicher Spektren beschreiben.

Diese Diskretisierung ignoriert, wie sich das Emissionsspektrum einer Lichtquelle durch Interaktionen mit der Szene verändert, und sorgt somit für verfälschte Farben. Zudem kann so keine wellenlängenabhängige Pfadgeometrie wie etwa durch Lichtbrechung oder variierende Pfadlängen in einem Medium simuliert werden.

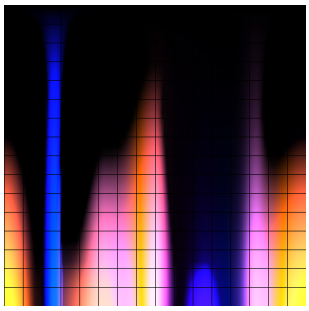
Spektrales Rendering simuliert die Reflektion von Licht an Oberflächen abhängig von dessen Wellenlänge. Dafür müssen RGB-Eingabedaten mit Hilfe von *spektralem Upsamling* in (Reflektanz-)Spektren umgewandelt werden. Um solche Spektren eindeutig zu erzeugen entwickelten Jakob und Hanika [6] einen Funktionsraum mit drei Parametern, basierend auf den Eigenschaften realer Reflektanzen. Diese Parameter werden für gegebene RGB-Tripel so angepasst, dass das resultierende Spektrum deren Farbe entspricht.

In größeren Farbräumen existiert für besonders gesättigte RGB-Tripel kein physikalisch realistisches (also von 0% und 100% Reflektanz begrenztes) Spektrum. Dafür wurde in der Arbeit [3] in Kooperation mit der Karlsuniversität Prag und EPFL Lausanne das Reflektanzmodell [6] um eine fluoreszente Komponente erweitert, um so gesättigtere, hellere Reflektanzen physikalisch plausibel abbilden zu können. Das Fluoreszenzmodell erfordert drei weitere Koeffizienten, welche gemeinsam mit den drei ursprünglichen Koeffizienten an Eingabe-RGB-Tripel angepasst werden. Dadurch erreichen wir einen höheren darstellbaren Gamut für das Upsampling großer Farbräume wie beispielsweise ACEScg. Die Abbildung unten zeigt ein Vielfaches der komponentenweisen absoluten Differenz zwischen der durch das Spektrum dargestellten RGB-Farbe und der Ziel-RGB-Farbe.

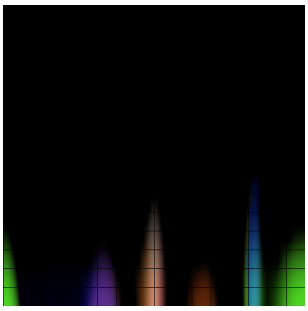
In einer darauf aufbauenden Arbeit [4] wurden weitere Eigenschaften der Modelle untersucht, sowie die Parametrisierung weiter verbessert, um Artefakten bei der Interpolation der nicht-linearen Reflektanzkoeffizienten entgegenzuwirken.



gesättigte Textur



Rekonstruktionsfehler ohne Fluoreszenz



Rekonstruktionsfehler mit Fluoreszenz

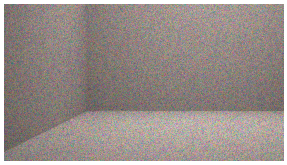
Welch's t-test zur Verifikation der Implementierung von Render-Algorithmen

Die korrekte Entwicklung und Formulierung von Monte Carlo-Bildsynthese-Verfahren ist herausfordernd und hat viele mögliche Fehlerquellen, sowohl in der Theorie als auch in der Umsetzung. Um die Korrektheit neuer Verfahren zu testen, wird deren Ausgabe oft mit einer Referenzimplementierung verglichen. Dies geschieht üblicherweise mithilfe des mittleren Bildfehlers (RMSE), eines Differenzbildes oder durch einen qualifizierten subjektiven Vergleich der entstandenen Bilder. Solche Methoden sind oft langwierig, da eventuelle Fehler initial durch starkes Rauschen überdeckt sein können, welches erst nach längerer Berechnungszeit, also mit vielen Lichttransportpfaden, verschwindet.

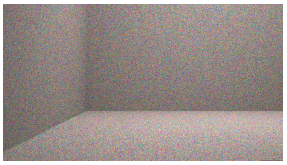
Welch's t-Test [7] ist ein statistischer Test. Gegeben zwei Mengen von Zufallszahlen, die aus zwei Normalverteilungen gezogen wurden, untersucht er, ob die Verteilungen den selben Mittelwert haben. Die Mengen selbst und die Varianz der Verteilungen können verschieden groß sein. Das Ergebnis ist ein p-Wert, welcher die Konfidenz über Gleichheit der Mittelwerte ausdrückt.

Die Arbeit [5] setzt Welch's t-Test ein, um zu testen, ob die Ausgaben von zwei Monte Carlo-Bildsynthese-Verfahren zum selben Ergebnis konvergieren. Dies offenbart systematische Fehler bereits mit wenigen Pfaden, also noch stark verrauschten Bildern.

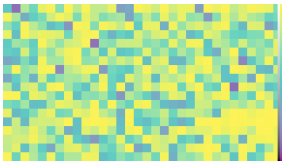
Dafür werden zunächst pro Iteration die Beiträge der Lichtpfade über einen Pixelblock (z.B. 32×32 Pixel) hinweg akkumuliert. Dadurch erhalten wir aus den stark nicht-normalverteilten Pfadbeiträgen individueller Pixel eine ungefähre Normalverteilung für den Block als Eingabe für den Test. Der Test wird dann in vielen Pixelblöcken parallel ausgeführt. Das ermöglicht die Identifikation besonders problematischer Bildbereiche, was Hinweise auf die Art des Fehlers geben kann. Zusätzlich kann ein Histogramm über die p-Werte aller Pixelblöcke ein verfälschtes Ergebnis selbst dann detektieren, falls dieses in einzelnen Blöcken nicht erkennbar ist.



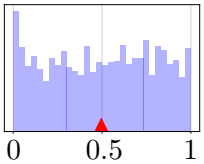
Referenz



fehlerhafte Impementierung



Welch's t-test (blockweise)



Histogramm: Spitze bei p≈0

2 Summary

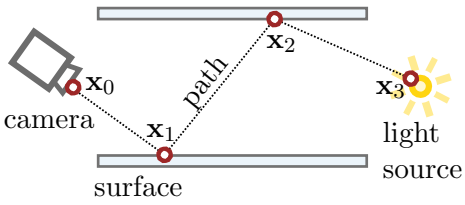
Photorealistic image synthesis (*rendering*) employs physically based light transport simulations to generate images from the virtual description of a scene. Its applications range from predictive rendering (e.g. for simple previews of new materials, light simulation for solar cell farms, generating training input for machine learning, or simulation of the energetic properties of buildings) over product or architecture visualization to the movie industry, where a multitude of different materials and light sources interact with each other. To this end numerical methods help us simulate radiative transport, or in other words how light propagates in a virtual scene. In this context the scene provides the parameters of the simulation, and the goal of the simulation is to approximate the light that arrives at a virtual camera sensor.

This thesis contributes to several phases of this simulation. In chapter 5 we investigate how to take input data in the form of RGB triples and transform them to fluorescent spectral data, so the light transport simulation can account for the wavelength dependent behavior of light. In chapters 6 and 7 we introduce two approaches that can help us handle particularly difficult scene configurations by introducing a small systematic error to the underlying integration problem in exchange for faster convergence and lower noise. Finally, in chapter 8 we propose a new method for quickly detecting systematic errors in the simulation without requiring fully converged results.

2.1 Introduction

The goal of photorealistic image synthesis is to approximately compute the incident light on a virtual camera sensor. The *rendering equation*

$$I_j = \int_{\mathcal{P}} f(\mathbf{X}) \, \mathrm{d}\mathbf{X}$$

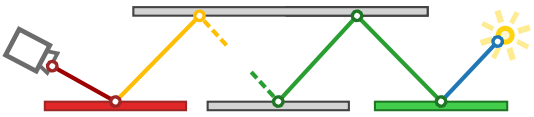


describes this incident light I_j at a camera pixel j as the integral over the contribution f of all light transport paths $\mathbf{X} = (\mathbf{x}_0, \ldots, \mathbf{x}_k)$ from the space $\mathcal{P}$ of all possible paths that photons can take to travel from a light source to the camera sensor. The contribution f includes factors for the intensity of the light source and the reflectance behavior of surface vertices $\mathbf{x}_i$, and depends on the light's wavelength λ.

Since the integral I_j is high dimensional, in practice we usually solve it with Monte Carlo integration, thanks to its favorable convergence properties: We stochastically draw N random and independent light transport paths, and then accumulate their weighted contributions. The variance of this approach manifests in noise in the image and vanishes with increasing sample size N.

A simple strategy for sampling such light transport paths is path tracing, where we start path construction on a sensor pixel j and then iteratively trace rays throughout the scene until we find a light source. Bidirectional path tracing ($BDPT$) is a more robust version of this, where we start one sub-path on the sensor and another at a light source, and connect the two sub-paths to form a full light transport path.

Either way some scene configurations introduce (near) singularities to the integrand, which are either hard to sample, thus requiring a big number of samples for a noise-free result, or cannot be sampled at all. *Mollification* can solve both of these issues by regularizing the integrand. With appropriate mollification parameters this leads to lower variance and shorter computation times, with only a low to negligible systematic error.



Fluorescence causes particularly bright and saturated appearances, as for instance in safety vests or text markers, by changing the wavelength of incident light. In the context of rendering this poses additional challenges, since accounting for a potential wavelength change at each scene vertex increases the integral's complexity. In the figure on the right the red and green surface are fluorescent and thus change the wavelength along the camera and light sub-path. This causes problems for BDPT, since the camera and light sub-paths may now end in different wavelengths.

2.2 Contributions

This thesis contributes to efficient and robust Monte Carlo image synthesis, with a focus on new approaches to mollification and fluorescent materials.

- The paper [1] investigates spectral regularization to enable connections between camera and light sub-paths that end in different wavelengths due to fluorescence. The paper [2] instead works on regularization for connections between specular surfaces (e.g. mirrors) and singular emitters (e.g. point lights), and aims to provide a larger framework for understanding (near-) singularities and their regularization in the rendering integral.

- The papers [3] and [4] reproduce the color impressions of RGB-triple inputs by combining reflectance and fluorescence spectra in order to obtain particularly saturated and intense colors in a physically reasonable context.

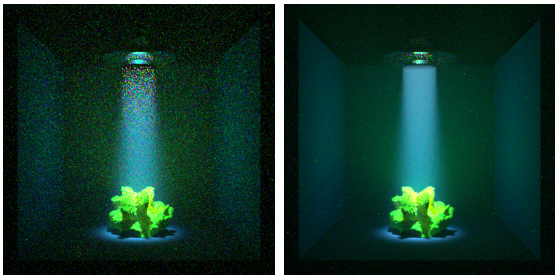
- The development and implementation of new rendering algorithms can be prone to errors that are hard to detect, even at high sample counts N. The paper [5] proposes a test to detect systematic errors in a light transport simulation from only a handful of samples.

Spectral Mollification for Bidirectional Fluorescence

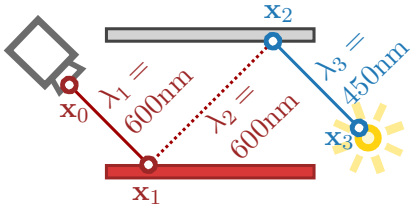
With spectral bidirectional path tracing (BDPT) we usually sample a random wavelength, for which we then construct and connect a camera and light sub-path. Fluorescent surfaces can change the wavelength along these paths (see the figure above) - their reflectance functions $f_f(\lambda_i, \lambda_o)$ are continuous and differentiable over the incident (λ_i) and exitant (λ_o) wavelength domain. Conventional reflectance models however are only defined for light leaving the surface with the same wavelength at which it arrived. We can represent this with a symbolic Dirac $f_r(\lambda_i, \lambda_o) = \delta_{\lambda_o}(\lambda_i) \cdot \dots$ as a singularity in the wavelength domain. Now, if we connect a camera and light sub-path that end in different wavelengths (as in the figure below), the contribution of this path is no longer well-defined.

In chapter 6 we mollify the singularity $\delta_{\lambda_o}(\lambda_i)$ of the non-fluorescent interaction at one of the connecting vertices, by replacing it with a family of functions that are easy to integrate and converge to the original singular behavior with $N \to \infty$ simulated paths. The mollified interaction evaluates to 0 if the incident and exitant wavelengths are more than d nm apart, and otherwise depends on the original reflectance. One good option (since it is simple, effective and energy conserving) is to evaluate the original reflectance for the incident wavelength and normalize the result with $(2d)^{-1}$.

After limited computation time the resulting systematic error produces darker colors and/or colors with a shifted hue. With appropriate mollification parameters (the maximum acceptable distance d between incident and exitant wavelength, as well as the rate at which d shrinks over iterations N) these errors are hard to impossible to detect with a human eye. For certain shrinking rates this method is consistent, and converges to the correct result with $N \to \infty$ paths while the error vanishes.



Path Tracer Mollified BDPT



Connecting a camera path $(\mathbf{x}_0, \mathbf{x}_1)$ with red to a light path $(\mathbf{x}_3, \mathbf{x}_2)$ with blue wavelength, mollification at $\mathbf{x}_2$.

Geometric Mollification for Spectral Next Event Estimation

Singularities in the rendering integrand cause problems for many Monte Carlo path tracing variants, since a continually sampled random number can never match a discrete singularity. Similarly, near-singularities (meaning integrands with most of their mass over a tiny support) can be hard to sample if we have to hit the support with a random number that was sampled from some unrelated sampling strategy. For example, conventional path tracing samples continuous directions at scene surfaces, and as such is completely

unable to hit a point light, and only rarely hits small area light sources. Next event estimation (explicitly connecting a vertex on a scene surface to a vertex on a light source) solves this issue, but in turn fails for (near) singular surfaces such as mirrors or glass. If singular surfaces and light sources appear together, both of these approaches fail.

In chapter 7 we investigate different representations of such singularities and derive a new regularization technique for next event estimation from singular surfaces and/or to singular light sources. We further discuss the relationship between kernel density estimation and regularization, and evaluate how the relative size of the kernel and a near-singularity's support influences variance and bias.

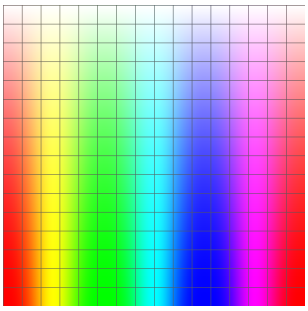
Wide Gamut Spectral Upsampling with Fluorescence

In practice, many light transport simulations not only disregard fluorescence, but ignore wavelength-dependent behavior altogether and do not even resolve their input data spectrally. Instead they compute the contribution of a path separately for a red, green and blue channel. Thus the materials used to model a scene are usually provided in RGB or other tristimulus color spaces that describe color impressions using three discrete values instead of continuous spectra.

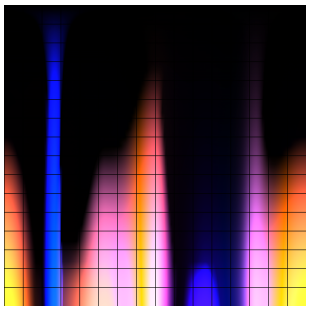
This simplification ignores how the emission spectrum of a light source and the reflectance spectra along a light transport path interact. This causes incorrect indirect illumination from paths with multiple interactions between the light source and camera, and ignores wavelength-dependent path geometry, e.g. from refraction or varying path lengths in a medium.

Spectral rendering simulates and evaluates light-surface-interactions depending on the wavelength. We thus need to convert RGB input data to (reflectance) spectra via *spectral upsampling*. This process is ambiguous, since infinitely many spectra correspond to the same RGB triple under a given illuminant. Based on the properties of real-world reflectances, Jakob and Hanika [6] design a function space based on three parameters that unambiguously produces such spectra. These parameters are fitted to any given RGB triple such that the corresponding spectra reproduces the desired RGB color impression.

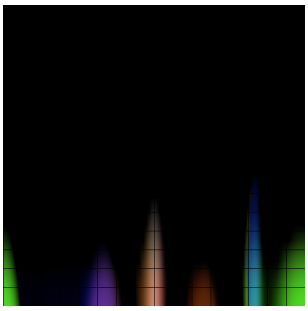
In bigger color spaces some RGB triples are too intense to be reproduced by a physically valid reflectance spectrum, i.e. a spectrum between 0% and 100% reflectivity. In cooperation with Charles University Prague and EPFL Lausanne we extend the model by [8] with a fluorescent component to achieve more saturated and intense reflectivities in a physically plausible context (chapter 5). Our fluorescence model needs three additional parameters, and we jointly fit our parameters and the three reflectance spectrum parameters to input RGB triples. The resulting gamut is larger than if we were restricted to reflectance spectra without fluorescence and is thus suitable for uplifting of large color spaces such as ACEScg. The figure below shows the component-wise absolute difference of target RGB triples and the closest possible RGB triple that can be reproduced with and without fluorescence.



Saturated texture



Reconstruction error without fluorescence



Reconstruction error with fluorescence

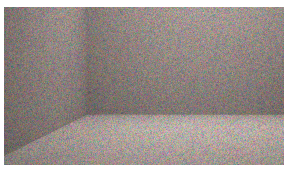
Detecting Bias in Monte Carlo Renderers using Welch's t-test

Developing and implementing new Monte Carlo image synthesis algorithms is challenging and prone to errors, both in the underlying theory and the actual code. In practice, new implementations are usually tested for correctness by comparing their output (i.e. a rendered image) to the output of a reference implementation, for instance by considering the average image error (RMSE), a difference image, or an informed subjective comparison of the two images. All of these approaches take time, as they require both a converged reference image, and a fairly converged test image, since potential error may be hidden behind initially strong noise that only vanishes after sampling a large number of light transport paths.

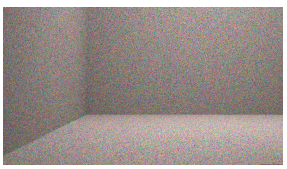
Welch's t-test is a statistical test. It takes two sets of random numbers from two normal distributions as input, and it outputs a p-value to express how confident we can be that the two normal distributions have the same mean. The two sets can have different sizes, and the variances of the two normal distributions do not have to be equal.

In chapter 8 we employ Welch's t-test to test if the outputs of two Monte Carlo renderers converge to the same result. This reveals systematic error after rendering only a handful of light transport paths, when the images still exhibit significant noise.

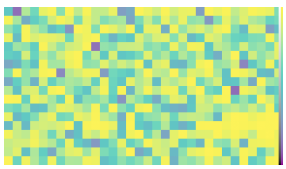
In each iteration we accumulate the contributions of all light transport paths per pixel block (e.g. 32×32 pixels). This transforms the typically non-normal distributed path contributions to roughly normal distributed input samples for Welch's t-test. We then perform the test in parallel for all of the pixel blocks, enabling us to identify particularly problematic image regions. In addition we visualize a histogram of the p-values of all pixel blocks, to detect bias even if individual pixel blocks seem acceptable in isolation.



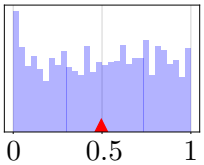
Reference



Biased Implementation



Welch's t-test (block wise)



Histogram: Peak at p≈0

3 List of publications

Publications included in this dissertation

A. Jung, A. Wilkie, J. Hanika, W. Jakob, and C. Dachsbacher (2019) "Wide Gamut Spectral Upsampling with Fluorescence", In: *Computer Graphics Forum* 38.4 [3]

L. König, A. Jung, and C. Dachsbacher (2020) "Improving Spectral Upsampling with Fluorescence", In: *Workshop on Material Appearance Modeling* (R. Klein and H. Rushmeier, eds.), The Eurographics Association [4]

A. Jung, J. Hanika, and C. Dachsbacher (2020) "Spectral Mollification for Bidirectional Fluorescence", In: *Computer Graphics Forum* 39.2 [1]

A. Jung, J. Hanika, and C. Dachsbacher (2020) "Detecting Bias in Monte Carlo Renderers using Welch's t-test", In: *Journal of Computer Graphics Techniques (JCGT)* 9.2 [5]

A. Jung and C. Dachsbacher (2022) "Spatial Mollification for Specular Next Event Estimation", unpublished [2]

Additional Publications

A. Jung, J. Hanika, S. Marschner, and C. Dachsbacher (2018) "A Simple Diffuse Fluorescent BBRRDF Model", In: *Workshop on Material Appearance Modeling* (R. Klein and H. Rushmeier, eds.), The Eurographics Association [8]

F. Simon, A. Jung, J. Hanika, and C. Dachsbacher (2018) "Selective Guided Sampling with Complete Light Transport Paths", In: *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 37.6 [9]

V. Schüßler, J. Hanika, A. Jung, and C. Dachsbacher (2022) "Path guiding with vertex triplet distributions", In: *Computer Graphics Forum* 41.4 [10]

E. Hamann, A. Jung, and C. Dachsbacher (2023) "Guiding Light Trees for Many-Light Direct Illumination", In: *Eurographics 2023 - Short Papers* (V. Babaei and M. Skouras, eds.), The Eurographics Association [11]

4 Background

This chapter covers the mathematical and physical foundations for this thesis. We start with the mathematical foundations of measures and integration (section 4.1), probability theory (section 4.2) and Monte Carlo integration (section 4.3).

Next we introduce the physical concepts behind photorealistic image synthesis in section 4.4, and cover a selection of common numerical procedures, so-called "rendering algorithms", for solving the corresponding equations in section 4.5. We extend these concepts to the wavelength dependency of light-matter-interaction in section 4.6, including wavelength-dependent radiometric properties, common numerical solutions and representation of spectral input data. Section 4.7 covers fluorescence, a particular spectral effect, where we discuss the physical backgrounds followed by existing methods for modeling and simulating fluorescence for photorealistic image synthesis.

In section 4.8 we present mollification, an essential tool for improving existing numerical solutions in the presence of (near) singular light transport phenomena such as mirrors or point light sources. Lastly, we discuss methods for evaluating and verifying the correctness of the output of a rendering algorithm in section 4.9.

Some parts of this chapter are taken from the publications [3], [4], [1], [2] and [5].

4.1 Measures and Integration

This section discusses measures in the context of integration, following Otto Forster's book on analysis [12]. We start in section 4.1.1 by defining what a measure is, what it means for a function to be measurable, and how to integrate a measurable function with respect to some measure. In section 4.1.2 we then show how to change the integration domain and how this affects the measure.

4.1.1 Definitions

This section clarifies what we mean by an integral of a real-valued function $f : \Omega \to \mathbb{R}$ over some domain Ω with a measure μ, as in

$$\int_\Omega f(u)\,\mathrm{d}\mu(u).$$

The measure. Intuitively, a measure is a function that "measures" sets. In the context of integration, we can interpret the measure μ of an integral as a weight for how much f contributes to the integral in different regions of the integration domain Ω. A measure is a function $\mu : \mathcal{D} \to \mathbb{R}_0^+$ where

1. $\mathcal{D}$ is a σ-Algebra on Ω,

2. $\mu(\emptyset) = 0$, and
3. $\mu(\bigcup_{i=1}^{\infty} A_i) = \sum_{k=1}^{\infty} \mu(A_i)$ for each sequence of pairwise disjoint sets $A_k \in \mathcal{D}, k \geq 1$.

Two important measures are the *Dirac measure*

$$\delta_x(A) := \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{else,} \end{cases} \tag{4.1.1}$$

and the *Lebesgue measure* λ^n on the Borel σ-algebra $\mathcal{B}(\mathbb{R}^n) \subseteq \mathcal{P}(\mathbb{R}^n)$

$$\lambda^n(A) := Vol(A)$$

corresponding to the volume of each set $A \in \mathcal{B}(\mathbb{R}^n)$. Since an integral with Lebesgue measure behaves equivalent to a Riemann integral under "reasonable" circumstances, we sometimes abbreviate such integrals as

$$\int_{\mathbb{R}^n} f(u)\,\mathrm{d}u := \int_{\mathbb{R}^n} f(u)\,\mathrm{d}\lambda^n(u).$$

The integral for simple functions. We first define integrals for so-called *simple functions*, and then extend the definition to arbitrary measurable functions.

Simple functions are functions that only take on a finite set of values $a_1, \ldots, a_n$, with disjoint supporting regions $A_i := \{u \in \Omega \mid f(u) = a_i\}$ per value. We can write them as

$$f = \sum_{i=1}^{n} \chi_{A_i} a_i,$$

using the characteristic function

$$\chi_A : \Omega \to \mathbb{R}$$
$$\chi_A(u) = \begin{cases} 1 & \text{if } u \in A \\ 0 & \text{else.} \end{cases}$$

The integral of a simple function with respect to measure $\mu : \mathcal{D} \to \mathbb{R}_0^+$ is defined as

$$\int_\Omega \sum_{i=1}^{n} \chi_{A_i}(u) a_i \,\mathrm{d}\mu(u) := \sum_{i=1}^{n} a_i \mu(A_i).$$

We can interpret this as "measuring" the size of each subset A_i with measure μ, to use as a weight for the constant function value a_i.

The integral for non-negative measurable functions. We now extend the integral definition to non-negative measurable functions $f : \Omega \to \mathbb{R}_+$, i.e. functions where

$$\forall B \in \mathcal{B}(\mathbb{R}) : f^{-1}(B) = \{u \in \Omega : f(u) \in B\} \in \mathcal{D}.$$

For each such function f there exists an ascending sequence $0 \leq f_1 \leq f_2 \leq \ldots$ of simple functions $f_k : \Omega \to \mathbb{R}_+$ that converges to f pointwise and monotonic [12, §4 Satz 5 (p. 43), p. 46]. The integral of f is then defined based on any such sequence as

$$\int_\Omega f(u)\,\mathrm{d}\mu(u) := \lim_{k\to\infty} \int_\Omega f_k(u)\,\mathrm{d}\mu(u).$$

Arbitrary integration domains and measurable functions. We further define the integral over a subset $A \subseteq \Omega$ as

$$\int_A f(u)\,\mathrm{d}\mu(u) := \int_\Omega \chi_A(u) f(u)\,\mathrm{d}\mu(u),$$

and the integral of a real-valued function $f : \Omega \to \mathbb{R}$ by splitting the support of f into two regions $S^+ := \{u \in \Omega \mid \ f(u) > 0\}$, $S^- := \{u \in \Omega \mid \ f(u) < 0\}$ and setting

$$\int_\Omega f(u)\,\mathrm{d}\mu(u) := \int_{S^+} f(u)\,\mathrm{d}\mu(u) - \int_{S^-} |f(u)|\,\mathrm{d}\mu(u),$$

if and only if the two positive integrals on the right-hand side have a finite value. More details on these steps can be found in [12].

4.1.2 Changing the Integration Domain

Under certain conditions we can change the integration domain of an integral of a measurable function $f : \Omega' \to \mathbb{R}$ (proofs in [12, ch. 6, Satz 4 and 5]). This transformation relies on a second function $\Phi : \Omega \to \Omega'$ describing how to warp arguments from a different domain Ω to the domain Ω' where f is defined. To reflect the distortion caused by Φ we need to adapt the measure, although in case of the Lebesgue measure we can extract the corresponding terms to the integrand.

Special case: Lebesgue measure. For open sets $\Omega, \Omega' \in \mathbb{R}^n$ and if Φ is $\mathcal{C}^{-1}$ invertible:

$$\int_\Omega f(\Phi(v))|\det D\Phi(v)|\,\mathrm{d}\lambda^n(v) = \int_{\Omega'} f(u)\,\mathrm{d}\lambda^n(u). \tag{4.1.2}$$

In this case the integrand changes from f to $f \circ \Phi|\det D\Phi|$ while the measure stays the same. The additional Jacobian determinant $|\det D\Phi|$ in the integrand on the left explicitly expresses the distortion caused by Φ. However, this rule is only applicable if the initial and target integration domain are (a subset of) $\mathbb{R}^n$, and if the measure is the Lebesgue measure.

General case: Arbitrary measure. For general integration domains Ω, Ω' with σ-algebras $\mathcal{D}, \mathcal{D}'$ and measure $\mu : \mathcal{D} \to \mathbb{R}_+$, we may instead use the following rule:

$$\int_\Omega f(\Phi(v))\,\mathrm{d}\mu(v) = \int_{\Omega'} f(u)\,\mathrm{d}\nu(u),$$

where $\nu := \Phi_*\mu : \mathcal{D}' \to \mathbb{R}_+$ is the pushforward measure of μ under Φ:

$$\Phi_*\mu(B) := \mu(\Phi^{-1}(B)) := \mu(\{a \in \mathcal{D} : \Phi(a) \in B\}). \tag{4.1.3}$$

In this case the integrand only changes from f to $f \circ \Phi$ and the Jacobian determinant is no longer present in either integrand. Instead, the pushforward measure ν implicitly accounts for the distortion caused by Φ.

The Lebesgue measure special case directly follows from the general case, since using the measure

$$\alpha(A) := \int_A |\det D\Phi(v)|\,\mathrm{d}\lambda^n(v) \tag{4.1.4}$$

we can rewrite the left-hand side of equation 4.1.2 as

$$\int_\Omega f(\Phi(v))|\det D\Phi(v)|\,\mathrm{d}\lambda^n(v) = \int_\Omega f(\Phi(v))\,\mathrm{d}\alpha(v),$$

and observe that λ^n is indeed the pushforward measure of α under Φ, since

$$\Phi_*\alpha(B) \overset{4.1.3}{=} \alpha(\Phi^{-1}(B)) \overset{4.1.4}{=} \int_{\Phi^{-1}(B)} |\det D\Phi(v)|\,\mathrm{d}\lambda^n(v) \overset{4.1.2}{=} \int_B 1\,\mathrm{d}\lambda^n(u) = \lambda^n(B).$$

Domain change for singularities. These rules only hold if the symbols used in the integrand are scalars and functions. In computer graphics, singular integrands are often expressed with the Dirac delta δ_a (see equation 4.8.3), as in

$$\int_{\Omega'} f(x)\delta_a(x)\,\mathrm{d}x := f(a).$$

This notation can be convenient for symbolically uniting multiple concepts (singularities and continuous behavior) in a single expression. However, such cases require particular attention to the integration domain and measure, as misinterpreting the symbolic δ_a as a function while changing domains may quickly lead to inconsistencies along the lines of

$$\int_{\Omega'} f(x)\delta_a(x)\,\mathrm{d}x = \int_\Omega f(\Phi(v))\delta_a(\Phi(v))|\det D\Phi(v)|\,\mathrm{d}v = |\det D\Phi(\Phi^{-1}(a))|f(a) \neq f(a).$$

We discuss this further in section 4.8, and deal with issues caused by multiple interacting singularities in different measure spaces in chapter 7.

4.1.3 Examples

We conclude the section on integration and measure with three integration domains that we will encounter later, along with a dedicated measure for each domain, and present the rules for switching between them.

The solid angle measure. Integrals with solid angle measure σ

$$\int_{\mathbb{S}^2} f(\omega)\,\mathrm{d}\sigma(\omega)$$

integrate a function $f : \mathbb{S}^2 \to \mathbb{R}$ over the unit sphere $\mathbb{S}^2$ with equal weight everywhere on the surface of $\mathbb{S}^2$. The solid angle measure computes the solid angle (in other words, the area or size) of surface patches $D \subseteq \mathbb{S}^2$ on the unit sphere, similar to how the Lebesgue measure λ^2 computes the area of surface patches on $\mathbb{R}^2$. We use the symbols $\mathbb{S}^2$ and σ independently of the actual parameterization of the unit sphere, since the size of a surface patch on the unit sphere is the same no matter how it is parameterized.

One example for such a parameterization are spherical coordinates

$$\mathbb{D}^2{}_{\theta\phi} := \{(\theta, \phi) \in [0, \pi] \times [0, 2\pi]\},$$

as in figure 4.1. We can compute the solid angle of a patch $D \subseteq \mathbb{D}^2{}_{\theta\phi}$ as

$$\sigma(D) = \int_0^{2\pi} \int_0^{\pi} \chi_D \sin\theta\,\mathrm{d}\lambda^2(\theta, \phi).$$

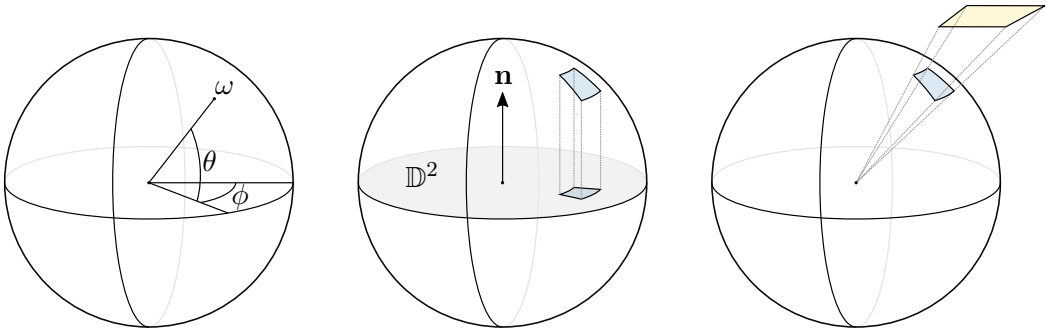


Figure 4.1: Spherical coordinates (θ, ϕ) of a direction $\omega \in \mathbb{S}^2$ (**left**), solid angle and projected solid angle (**center**), and solid angle of a surface patch (**right**).

This results in the following formula for a solid angle integral:

$$\int_{\mathbb{S}^2} f(\omega)\,\mathrm{d}\sigma(\omega) = \int_0^{2\pi}\int_0^{\pi} f(\theta,\phi)\sin\theta\,\mathrm{d}\lambda^2(\theta,\phi).$$

The projected solid angle measure. Sometimes it is convenient to change the integration domain by projecting the unit sphere $\mathbb{S}^2$ onto a unit disk $\mathbb{D}^2$ with normal $\mathbf{n}$, as in figure 4.1. We use the projected solid angle measure $\sigma^\perp$ to integrate the unit disk $\mathbb{D}^2$ with equal weight everywhere on the disk. As with the solid angle measure, we use the notations $\mathbb{D}^2$ and $\sigma^\perp$ independently of the actual parameterization, to refer to the concept of the unit disk as well as the size of some surface patch on $\mathbb{D}^2$, respectively.

Let $\Phi_{\sigma\to\sigma^\perp} : \mathbb{S}^2 \to \mathbb{D}^2$ be the mapping that projects directions ω from $\mathbb{S}^2$ to $\mathbb{D}^2$ along $\mathbf{n}$, and let $f : \mathbb{D}^2 \to \mathbb{R}$ be a function on the unit disk. The domain change restricted to the upper hemisphere $\mathbb{H}^2 = \{\omega \in \mathbb{S}^2 | \langle \mathbf{n}, \omega\rangle > 0\}$ works as follows:

$$\int_{\mathbb{D}^2} f(\omega^\perp)\,\mathrm{d}\sigma^\perp(\omega^\perp) = \int_{\mathbb{H}^2} f(\Phi_{\sigma\to\sigma^\perp}(\omega))\langle \mathbf{n}, \omega\rangle\,\mathrm{d}\sigma(\omega).$$

The cosine $\langle \mathbf{n}, \omega\rangle$ between projection direction $\mathbf{n}$ and ω accounts for the distortion from projecting a surface patch from the sphere down to the disk, as in figure 4.1.

For convenience we also define $f(\omega) := f(\Phi_{\sigma\to\sigma^\perp}(\omega))$, and from now on use arguments $\omega \in \mathbb{D}^2$ from the unit disk and $\omega \in \mathbb{S}^2$ from the unit sphere interchangeably.

The surface area measure. We use the surface area measure A to integrate over a set of two-dimensional surfaces $\mathcal{M} \subseteq \mathbb{R}^3$ in three-dimensional space. The surface area measure of a piece of surface $D \subseteq \mathcal{M}$ is simply its area.

To switch between integration over surfaces $\mathcal{M}$ and the unit sphere $\mathbb{S}^2$ centered at some point $\mathbf{x}$, we consider the solid angle of surface patches viewed from $\mathbf{x}$ (see figure 4.1). We define the following shorthand notation to talk about directions between points:

$$\mathbf{x} \to \mathbf{y} := \frac{\mathbf{y}-\mathbf{x}}{||\mathbf{y}-\mathbf{x}||}.$$

We map points $\mathbf{y}$ from 2D surfaces to the unit sphere centered at $\mathbf{x}$ via

$$\Phi_{A\to\sigma}(\mathbf{y}) := \frac{\mathbf{y}-\mathbf{x}}{||\mathbf{y}-\mathbf{x}||} = \mathbf{x} \to \mathbf{y}.$$

The Jacobian of this mapping includes the surface normal $\mathbf{n}_y$ at $\mathbf{y}$

$$|\det D\Phi_{A\to\sigma}| = \frac{\langle \mathbf{n}_y, \mathbf{y}\to\mathbf{x}\rangle}{||\mathbf{x}-\mathbf{y}||^2}$$

and accounts for the rotation between the surface and $\mathbf{y}\to\mathbf{x}$, and the distance from $\mathbf{x}$ to the surface, since the solid angle of a surface shrinks quadratically with its distance.

The mapping is not injective, since multiple regions of $\mathcal{M}$ might map to the same solid angle when viewed from $\mathbf{x}$. In this case we usually only care about the closest non-occluded part and use the visibility function

$$V(\mathbf{x},\mathbf{y}) = \begin{cases} 1 & \text{if } \mathbf{y} \text{ is visible from } \mathbf{x} \\ 0 & \text{if } \mathbf{y} \text{ is occluded by some closer surface} \end{cases}$$

to switch between solid angle measure and surface area measure via

$$\int_{\mathbb{S}^2} f(\omega)\,\mathrm{d}\sigma(\omega) = \int_{\mathcal{M}} f(\Phi(\mathbf{y}))V(\mathbf{x},\mathbf{y})\frac{\langle \mathbf{n}_\mathbf{y}, \mathbf{y}\to\mathbf{x}\rangle}{||\mathbf{x}-\mathbf{y}||^2}\,\mathrm{d}A(\mathbf{y}).$$

Summary. We write $f(\omega^\perp)$, $f(\omega)$ and $f(\mathbf{y})$ interchangeably with $f(\omega) := f(\Phi_{\sigma\to\sigma^\perp}(\omega))$ and $f(\mathbf{y}) := f(\Phi_{A\to\sigma}(\mathbf{y}))$.

For points $\mathbf{x},\mathbf{y}\in\mathcal{M}$ with surface normals $\mathbf{n}_\mathbf{x}, \mathbf{n}_\mathbf{y}$ we can switch between integration over the unit disk at $\mathbf{x}$, unit sphere at $\mathbf{x}$ or visible surfaces via

$$\begin{aligned} &\int_{\mathbb{D}^2} f^+(\omega^\perp) + f^-(\omega^\perp)\,\mathrm{d}\sigma^\perp(\omega^\perp) \\ =&\int_{\mathbb{S}^2} f(\omega)|\langle \mathbf{n}_\mathbf{x}, \omega\rangle|\,\mathrm{d}\sigma(\omega) \\ =&\int_{\mathcal{M}} f(\mathbf{y})V(\mathbf{x},\mathbf{y})\frac{\langle \mathbf{n}_\mathbf{y}, \mathbf{y}\to\mathbf{x}\rangle|\langle \mathbf{n}_\mathbf{x}, \mathbf{x}\to\mathbf{y}\rangle|}{||\mathbf{x}-\mathbf{y}||^2}\,\mathrm{d}A(\mathbf{y}), \end{aligned} \tag{4.1.5}$$

with f^+ and f^- as the upper and lower hemisphere components of f with respect to $\mathbf{n}_\mathbf{x}$.

In particular, the solid angle measure at $\mathbf{x}$ and the surface area measure are linked via

$$\sigma(\Phi_{A\to\sigma}(D)) = \int_D V(\mathbf{x},\mathbf{y})\frac{\langle \mathbf{x}-\mathbf{y}, \mathbf{n}_\mathbf{y}\rangle}{||\mathbf{x}-\mathbf{y}||^2}\,\mathrm{d}A(\mathbf{y}),$$

and the projected solid angle measure is linked to the solid angle measure via

$$\sigma^\perp(\Phi_{\sigma\to\sigma^\perp}(D)) = \int_D \langle \omega, \mathbf{n}_\mathbf{x}\rangle\,\mathrm{d}\sigma(\omega).$$

4.2 Probability Theory

This section covers the basics of probability theory that we need for numerical integration in section 4.3, and defines the notation and terms used throughout this thesis.

4.2.1 Random Variables

In this section we introduce a mathematical framework for describing random processes, following Georgii's introduction to probability theory and stochastics [13], including

- probability mass and density functions on discrete and continuous event spaces,
- random variables as mappings between event spaces that can be interpreted as something whose value depends on the outcome of a random process, and
- the combination of the two concepts into the distribution[1] of a random variable.

Sample space and event space. Each random process has a set Ω of possible outcomes. This set can be countable or uncountable, and we call it the *sample space*. Combining a sample space with a σ-algebra $\mathcal{F}$ on Ω leads to the *event space* $(\Omega, \mathcal{F})$.

Probability space. Each subset of outcomes $A \in \mathcal{F}$ occurs with a certain probability. This probability can be expressed by a normalized measure $P : \mathcal{F} \to [0, 1]$, i.e.

1. P is normalized: $P(\Omega) = 1$
2. P is σ-additive: $P(\bigcup_{i\geq 1} A_i) = \sum_{i\geq 1} P(A_i)$ for all pairwise disjoint $A_1, A_2, \cdots \in \mathcal{F}$

We call P a *probability measure*, and the triple $(\Omega, \mathcal{F}, P)$ a *probability space*. Note that P takes subsets $A \subseteq \Omega$ of possible outcomes as input, not individual outcomes $\omega \in \Omega$.

Probability mass function. If Ω is countable, the σ-additivity of P implies that P can be constructed from a function $p : \Omega \to [0, 1]$ by setting

$$P(\{\omega\}) := p(\omega), \tag{4.2.1}$$

since the value of all subsets with more than one element follows directly from the σ-additivity. In this case p represents the probability of individual outcomes and we call it the *probability mass function (PMF)* of P.

Probability density function. If Ω is uncountable but a Borel set[2], i.e. $\Omega \in \mathcal{B}(\mathbb{R}^n)$, many (but not all) probability measures can be constructed from a function $p : \Omega \to \mathbb{R}^+$ where

- $\{\omega \in \Omega : p(\omega) \leq c\} \in \mathcal{B}(\mathbb{R}^n)$ for each $c > 0$
- p is normalized: $\int_\Omega p(\omega)\, \mathrm{d}\lambda(\omega) = 1$

[1] In section 4.8.2 we introduce the concept of "distributions" that map functions to scalars, e.g. by integrating them against another function. From now on, we use the term distributions to refer to distributions of random variables as introduced on the following pages, unless stated otherwise.

[2] Since the Borel algebra $\mathcal{B}(\mathbb{R}^n)$ includes all open and closed sets in $\mathbb{R}^n$ and their countable unions and intersections, this covers all domains that we encounter in later chapters.

via

$$P(A) := \int_A p(\omega)\,\mathrm{d}\lambda^n(\omega) \text{ for } A \in \mathcal{B}(\Omega), \tag{4.2.2}$$

where $\mathcal{B}(\Omega) = \{A \cap \Omega : A \in \mathcal{B}(\mathbb{R}^n)\}$ is the Borel σ-algebra on Ω. We call this p the *probability density function (PDF)* of P. Each PDF defines a probability measure, but not every probability measure has such a corresponding PDF.

Considering that probability measures are special measures, we can apply the rules from section 4.1.2 to express the same PDF over different domains or with respect to different measures, while conserving the normalization property. For example, we can convert a PDF $p_{\sigma^\perp}$ on the unit disk with normal $\mathbf{n}$ to a solid angle PDF p_σ on the unit hemisphere via

$$p_\sigma(\omega) = \langle \mathbf{n}, \omega \rangle \cdot p_{\sigma^\perp}(\omega),$$

and further convert from a solid angle PDF at $\mathbf{x}$ to a PDF p_A over surfaces via

$$p_A(\mathbf{y}) = V(\mathbf{x}, \mathbf{y}) \cdot \frac{\langle \mathbf{n}_\mathbf{y}, \mathbf{y} \to \mathbf{x} \rangle}{||\mathbf{x} - \mathbf{y}||^2} \cdot p_\sigma(\mathbf{x} \to \mathbf{y}). \tag{4.2.3}$$

Random variables. A random variable is a mapping between two event spaces $(\Omega, \mathcal{F})$ and $(\Omega', \mathcal{F}')$. Following [13], we define a random variable as a measurable function $X : \Omega \to \Omega'$, i.e. a function where

$$\forall B \in \mathcal{F}' : \{\omega \in \Omega : X(\omega) \in B\} =: X^{-1}(B) \in \mathcal{F}.$$

Distribution of a random variable. Let $(\Omega, \mathcal{F}, P)$ be a probability space and $(\Omega', \mathcal{F}')$ an event space, with $X : \Omega \to \Omega'$ as a random variable between the two spaces. Based on P and X we can construct a new probability measure P' on Ω' via

$$P'(A') := P \circ X^{-1}(A') = P(\{\omega | X(\omega) \in A'\}). \tag{4.2.4}$$

We call this P' the *distribution of X under P*. Intuitively speaking, P' warps its input A' back through X^{-1} to a set in Ω and then applies the existing probability measure P. We also use the short-hand notation

$$P(\{X \in A'\}) := P'(A')$$

to emphasize that $P'(A')$ is the probability measure P of the set $\{X \in A'\} := \{\omega \mid X(\omega) \in A'\}$, and further abbreviate this to

$$P(X \in A') := P(\{X \in A'\})$$

to support the point of view that realizations of the random variable take on values corresponding to the outcome of some random process. The notation $P(X \in A')$ then represents the probability of getting an outcome within a set A'.

To summarize, we can view

1. X as a function in line with definition 4.2.4, where $P(X \in A')$ is the probability that a random sample $\omega \in \Omega$ gets mapped to A' via $X(\omega)$.
2. the expression $P(X \in A')$ as the probability of the statement $X(\omega) \in A'$ for a random sample ω, or in other words, as the probability that a realization of the random variable X is in A'.

If P' can be constructed from some probability density or mass function p' as in equation 4.2.2, we say X *is distributed according to* p' or *the PDF of* X *is* p' and write

$$X \propto p'.$$

Two random variables with the same distribution P' are called *identically distributed*. Each probability measure P is also the distribution P' of the random variable $X : \Omega \to \Omega, \omega \mapsto Id(\omega) = \omega$, as in this case $X^{-1}(A') = A$ and thus $P'(A') = P(X^{-1}(A')) = P(A')$.

In later sections we often start from the *primary sample space* $\Omega = [0,1]^n$ with a probability measure P based on a uniform probability density function $p \equiv 1$. We then warp samples ω from Ω to Ω' using the *sampling procedure* X. The resulting sample $X(\omega)$ is distributed according to some PDF p' that depends on the sampling procedure.

Important probability density functions. Throughout this thesis we will encounter several specific PDFs. Important examples include

- The uniform PDF, which is constant over a bounded domain with equal density for all regions in it, such as the uniform PDF over the unit hypercube $[0,1]^n$

$$\mathcal{U}^{(n)}(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \in [0,1]^n \\ 0 & \text{else,} \end{cases}$$

  the uniform PDF over a one-dimensional interval $[a,b]$:

$$p(x) = \begin{cases} 1/(b-a) & \text{if } x \in [a,b] \\ 0 & \text{else,} \end{cases}$$

  and the uniform PDF over a two-dimensional surface with area A:

$$p(\mathbf{x}) = \begin{cases} 1/A & \text{if } \mathbf{x} \text{ on surface} \\ 0 & \text{else.} \end{cases}$$

  Two special cases of the latter are the uniform PDF over the unit disk:

$$p(\mathbf{x}) = \begin{cases} \frac{1}{\pi} & \text{if } \mathbf{x} \in \mathbb{D}^2 \\ 0 & \text{else,} \end{cases}$$

  and the uniform PDF over the unit hemisphere:

$$p(\mathbf{x}) = \begin{cases} \frac{1}{2\pi} & \text{if } \mathbf{x} \in \mathbb{S}^2 \\ 0 & \text{else.} \end{cases}$$

  Similarly, the uniform PMF on a finite set Ω assigns the same probability to each element:

$$p(\omega) = 1/|\Omega|.$$

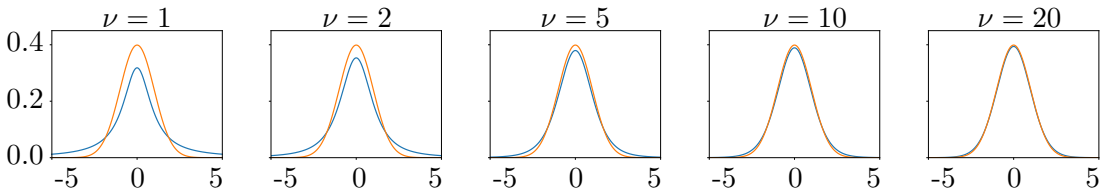


Figure 4.2: The t-distribution for $\nu = 1, 2, 5, 10, 20$ (blue) and the standard normal distribution $\mathcal{N}_{0,1}$ (orange). As $\nu \to \infty$ the t-distribution approaches $\mathcal{N}_{0,1}$.

- The normal distribution

$$\mathcal{N}_{\mu,\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}$$

  with mean μ and standard deviation σ. We say a random variable is *normally distributed* if it is distributed according to $\mathcal{N}$.

- The t-distribution

$$f_\nu(t) = \frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\Gamma(\nu/2)}(1 + \frac{t^2}{\nu})^{-\frac{\nu+1}{2}}$$

  with so-called degrees of freedom $\nu \in \mathbb{N}$, and Γ as the Gamma function[3]. With increasing degrees of freedom $\nu \to \infty$, the t-distribution converges to the standard normal distribution, as shown in figure 4.2:

$$f_\infty(t) := \lim_{\nu\to\infty} f_\nu(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} = \mathcal{N}_{0,1}(t)$$

Singularities. Some random processes in a continuous context only ever produce a single value or a discrete set of values. Or, in terms of the definitions from above, sometimes Ω and Ω' are uncountable, but $X(\Omega)$ is countable and maybe even finite. In this case, there may exist no function p' fulfilling the requirements to be the PDF of P'. For instance, consider $\Omega = \Omega' = [0, 1]$, P based on a uniform PDF and $X(\omega) \equiv 0.5$, in which case $P'(B) = \chi_B(0.5)$ does not have a PDF.

One way to deal with this is to simply restrict Ω' to the (discrete) image of X, in which case P' does have a probability *mass* function as in equation 4.2.1. Another option are symbolic notations like $p = \delta_a$ (or a sum of such deltas if $X(\Omega)$ contains more than one element), meant to describe that all the probability density concentrates in a single point. This may allow combining continuous and discrete cases in a single expression. However, this is a purely symbolic representation and not actually a function: For evaluation it is usually replaced by $p(a) = 1, p(x \neq a) = 0$, which does not integrate to 1 with "continuous" measures[4]. In addition, it potentially needs to be scaled by additional factors depending on the domain and measure to which such a density should have been normalized, if it existed in the first place.

[3]The Gamma function is a generalization of the factorial, i.e. for all integers $k \in \mathbb{N}$ it evaluates to $\Gamma(k) = (k-1)!$, and extends to real numbers in a smooth curve.

[4]The function $p(a) = 1, p(x \neq a) = 0$ does integrate to one when integrated against the Dirac measure δ_a from equation 4.1.1. However, in an equation that combines both continuous and discrete behavior in an integral, we usually want some sort of continuity in the measure to capture the continuous behavior as well, in which case this function integrates to 0 and is no longer a density.

4.2.2 Expected Value and Variance

Expected Value

The *expected value* of a real-valued random variable $X : \Omega \to \mathbb{R}$ describes the outcome we can expect, on average, from the random process behind the random variable. This average outcome is not necessarily in the image of the random variable.

The expected value of a discrete random variable X that is distributed according to some PMF P and can take on values x_i is defined as the sum of all possible values, weighted by their respective PMF:

$$\mathbb{E}(X) = \sum x_i \cdot P(x_i).$$

The expected value of a continuous random variable X that is distributed according to some PDF p and can take on values x in some domain Ω is

$$\mathbb{E}(X) = \int_\Omega x \cdot p(x) \, \mathrm{d}\mu(x),$$

where μ is the measure on Ω to which p is normalized.

In both cases, the expected value scales linearly with the random variable:

$$\mathbb{E}(aX) = a\mathbb{E}(X), \tag{4.2.5}$$

and for any constant a or random variable Y we can extract sums from within the argument as

$$\begin{aligned} \mathbb{E}(X + a) &= \mathbb{E}(X) + \mathbb{E}(a) \\ \mathbb{E}(X + Y) &= \mathbb{E}(X) + \mathbb{E}(Y). \end{aligned} \tag{4.2.6}$$

Sample mean

The expected value of a random variable X only depends on the underlying PMF or PDF of the process, and the domain of possible values. In some cases it can be computed analytically, such as the expected value of a dice roll or the expected position of a uniformly random dart throw. It can also be estimated by performing a random process over and over again and observing an increasing number of realizations $\{X_1, \ldots, X_N\}$ of X called *samples*, and then averaging the results in the *sample mean*

$$\bar{X} = \frac{1}{N} \sum_{i=1}^{N} X_i. \tag{4.2.7}$$

The law of large numbers [13, ch. 5] dictates that the sample mean approaches the true expected value with increasing N:

$$\bar{X} \to \mathbb{E}(X) \text{ for } N \to \infty.$$

This is the basis for Monte Carlo integration (section 4.3).

Variance

While the expected value describes how a random process behaves on average, the variance describes how much the actual observations are expected to differ from the expected value. It is defined as

$$V(X) = \mathbb{E}\left((X - \mathbb{E}(X))^2\right).$$

Based on the properties of the expected value (equations 4.2.5 and 4.2.6), the variance can also be expressed as

$$\begin{aligned} V(X) = \mathbb{E}\left((X - \mathbb{E}(X))^2\right) &= \mathbb{E}\left(X^2 - 2X\mathbb{E}(X) + \mathbb{E}(X)^2\right) \\ &= \mathbb{E}(X^2) - 2\mathbb{E}(X\mathbb{E}(X)) + \mathbb{E}(X)^2 \qquad \text{equation 4.2.6} \\ &= \mathbb{E}(X^2) - 2\mathbb{E}(X)\mathbb{E}(X) + \mathbb{E}(X)^2 \qquad \text{equation 4.2.5} \\ &= \mathbb{E}(X^2) - \mathbb{E}(X)^2. \end{aligned}$$

For scalar constants $a \in \mathbb{R}$, this results in

$$V(aX) = a^2 V(X),$$

and for any constant a or *independent* random variable Y we can extract sums as

$$\begin{aligned} V(X + a) &= V(X) + V(a) \\ V(X + Y) &= V(X) + V(Y). \end{aligned}$$

Sample variance

Just as with the expected value, we can estimate the variance of a random process based on an observed sample set $\{X_1, \ldots, X_N\}$ as the *unbiased sample variance*

$$S^2 = \frac{1}{N-1}\sum_{i=1}^{N}(X_i - \bar{X})^2 = \frac{1}{N-1}\left(\sum_{i=1}^{N} X_i^2 - \frac{(\sum_{i=1}^{N} X_i)^2}{N}\right). \tag{4.2.8}$$

This estimate is only defined if there are at least $N = 2$ samples available, since with only one observation ($N = 1$) the sample X_1 would be equal to the sample mean and the sample variance would be meaningless.

4.2.3 Random Number Generation

Pseudorandom numbers. Since computers work deterministically (aside from concurrency issues), true randomness in computer programs is hard to achieve. Instead, we use so-called *pseudorandom number generators*. These work deterministically (at least in single-threaded environments) and generate a sequence of numbers that are uniformly distributed in $[0, 1)$, and in most aspects share or at least approximate properties of truly random numbers [14]. The Mersenne twister is such a pseudorandom number generator. We use it for rendering some of the images in chapter 8. It creates a deterministic sequence of 32 bit numbers by modifying and combining previous numbers. Thanks to its large period the sequence seems endless without repetition, at least for our purposes.

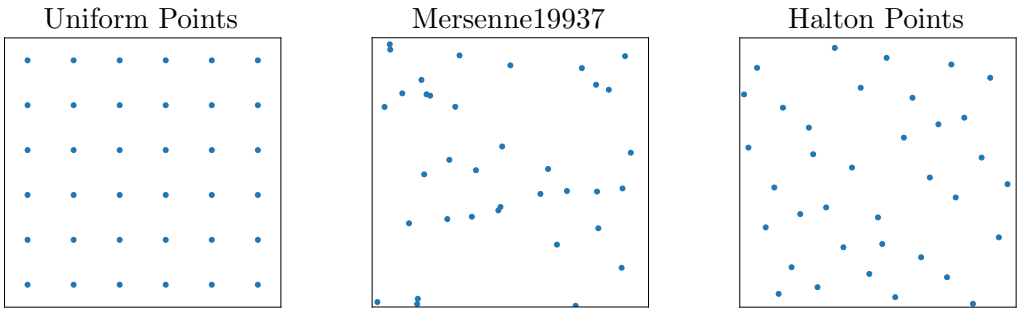


Figure 4.3: 36 Points evenly distributed (**left**), from the Mersenne19937 random number generator with seed 1234 (**center**), and the first 36 (2,3)-Halton points (**right**).

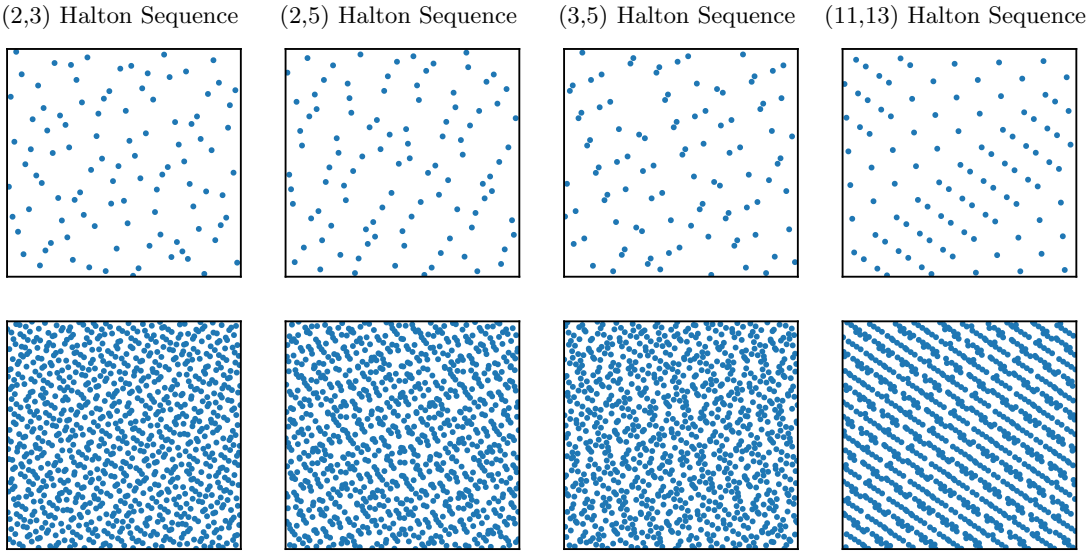


Figure 4.4: First 100 (**top**) and 1000 (**bottom**) 2D Halton points in the (2,3), (2,5), (3,5) and (11,13) Halton sequences. While all sequences have theoretically low discrepancy, some have more obvious structural clusters than others.

Quasirandom numbers. *Quasirandom numbers*, on the other hand, trade some aspects of randomness in favor of better stratification or discrepancy. They exhibit a certain level of correlation, sometimes to a different degree in different dimensions. For instance, *low-discrepancy sequences* are potentially infinite deterministic sequences of numbers with (prefixes of) low discrepancy, i.e. they distribute points as evenly as possible while still maintaining some appearance of randomness.

We use the *Halton sequence* for most renders in this thesis, as it distributes noise over an image in a way that is more agreeable to the human visual system than pseudorandom noise (see figures 4.3 and 4.5). Each Halton sequence is based on a prime number p, and by construction guarantees one point in each k/p bin after every p points. Combining sequences with different base primes offers good stratification in multidimensional sequences, although in higher dimensions the sequences starts exhibiting structural artifacts (figure 4.4) which can be alleviated by decorrelating dimensions through scrambling.

4.3 Monte Carlo Integration

Many of the integrals we encounter in the context of photorealistic image synthesis cannot reasonably be solved analytically. Instead, we have to rely on numerical integration. We discuss numerical integration in general and two important properties of numerical estimators, bias and consistency, in section 4.3.1. We then introduce the Monte Carlo estimator in section 4.3.2, and show how its quality can be improved through clever sampling strategies in sections 4.3.3 and 4.3.4.

4.3.1 Numerical Integration, Bias and Consistency

Given an integral

$$I = \int_\Omega f(x)\,\mathrm{d}\mu(x)$$

of some function f over some domain Ω, numerical integration methods evaluate the integrand f for a collection of N arguments $x_1, \ldots, x_N$ ("samples") and estimate the value of the integral in an estimator $F_N(f, x_1, \ldots, x_N)$ based on these sample values. Thus they can approximate integrals with no (simple) analytical solution, provided it is possible to evaluate the integrand f at the sample locations. Some approaches, such as the Simpson method, place their samples x_i deterministically, while others, such as Monte Carlo integration, rely on random sample locations.

Bias. The resulting estimator is *unbiased* if its expected value is equal to the value of the integral:

$$\mathbb{E}(F_N) = I$$

Otherwise it contains a systematic error b we call *bias*:

$$\mathbb{E}(F_N) = I + b.$$

In practice there exist different causes for bias. Bias can either occur by accident, if an implementation of an unbiased algorithm contains unintended errors. Or it can occur on purpose, if an estimator is designed to intentionally produce an incorrect value in exchange for other desirable properties, such as fast convergence to an almost correct result, or providing approximate solutions to otherwise completely unsolvable problems.

Chapter 8 proposes a new method to detect bias, both intentional and accidental, in Monte Carlo renderer implementations, and chapters 6 and 7 introduce bias to integration problems that would otherwise be hard or impossible to solve.

Consistency. Numerical integration methods usually improve the quality of their estimate with an increasing number N of samples x_i and evaluations $f(x_i)$. If an estimator converges to the correct integral value with increasing number of samples, we call the estimator *consistent*:

$$F_N \to I \text{ for } N \to \infty.$$

Unbiased estimators are always consistent. But so are some biased estimators, as long as the parameters that are responsible for the bias depend on the sample count, such that the systematic error b converges to zero with an increasing number of samples.

4.3.2 The Monte Carlo Estimator

The central integral in photorealistic image synthesis, the rendering integral (see section 4.4.6), is infinitely recursive, and even practical implementations are often quite high-dimensional. This makes integration approaches suffering from the curse of dimensionality unusable. Over the years, Monte Carlo methods emerged as the most popular choice for solving the rendering integral [15].

In contrast to numerical integration approaches that are based on deterministic sample locations, the variance of Monte Carlo integration only depends on the number of samples for which the integrand is evaluated, but not on the dimensionality of the integral. This makes it particularly suited to solve high-dimensional problems. Aside from that it is conceptually simple, and the randomized sample locations make it robust to some aliasing artifacts.

Let $f : \Omega \to \mathbb{R}_0^+$ be a real-valued function on some domain Ω and $p : \Omega \to \mathbb{R}_0^+$ be a probability density function on Ω that is normalized to the measure μ of the integral. Monte Carlo integration estimates the value I of an integral

$$I = \int_\Omega f(x) \, \mathrm{d}\mu(x)$$

in the Monte Carlo estimator

$$F_N = \frac{1}{N} \sum_{i=1}^{N} \frac{f(X_i)}{p(X_i)}$$

by averaging the integrand evaluated for N independent samples from p weighted by their respective PDF $p(X_i)$.

Variations of the Monte Carlo estimator in rendering include Multiple Importance Sampling (section 4.3.4), where we combine samples from multiple PDFs, Markov Chain Monte Carlo, where the samples are created by a Markov Chain, and quasi-Monte Carlo methods as in figure 4.5, where we use numbers from low-discrepancy sequences (see section 4.2.3) instead of pseudorandom numbers for subjectively smoother results.

Expected value

As long as the probability density p fulfills the condition

$$f(x) > 0 \Rightarrow p(x) > 0, \tag{4.3.1}$$

the expected value of the Monte Carlo estimator is equal to the true value of I:

$$\begin{aligned} \mathbb{E}(F_N) &= \mathbb{E}\left(\frac{1}{N} \sum_{i=1}^{N} \frac{f(X_i)}{p(X_i)} \right) = \mathbb{E}\left(\frac{f(X)}{p(X)} \right) \\ &= \int_\Omega \frac{f(x)}{p(x)} \cdot p(x) \, \mathrm{d}\mu(x) = \int_\Omega f(x) \, \mathrm{d}\mu x = I. \end{aligned} \tag{4.3.2}$$

The last step in equation 4.3.2 only works if p fulfills equation 4.3.1. In other words, we must be able to sample any value X_i that contributes to the integrand f.

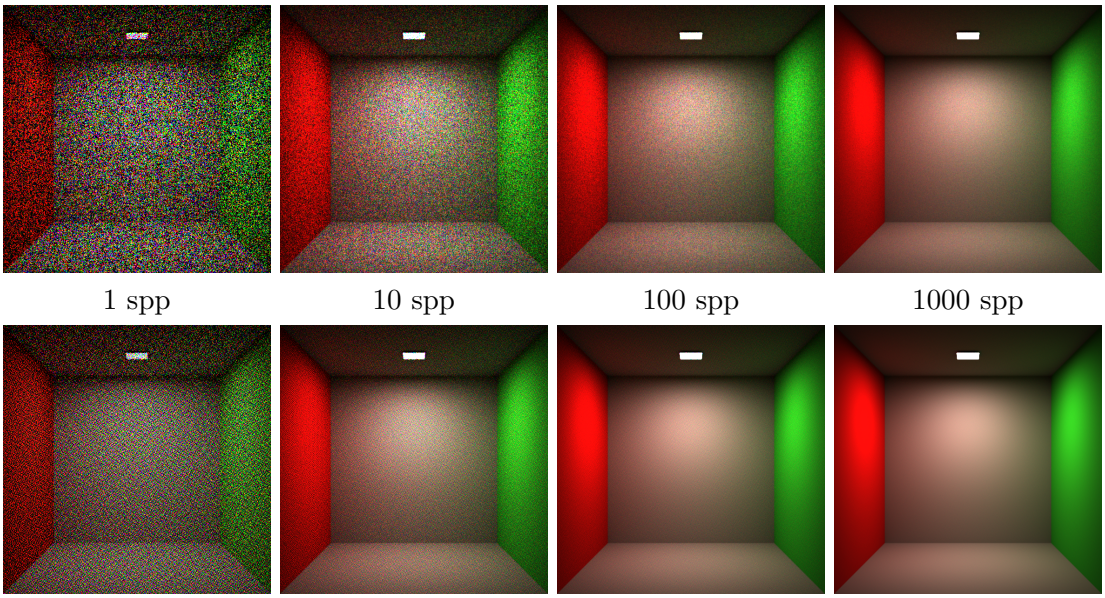


Figure 4.5: Quasi-Monte Carlo methods, while not truly random, can produce visually more pleasing results than Monte Carlo with pseudorandom numbers, as shown by these renders based on pseudorandom numbers (**top**) and Halton points (**bottom**).

If the condition in equation 4.3.1 does not hold, i.e. if the support of p does not cover the full support of f, then some regions where $f > 0$ are not sampled at all and the value of the estimator is biased (see section 4.3.1) to underestimate the true value.

Apart from that, we are free to choose p as we please. It turns out that good choices of p increase the rate by which F_N converges to I with additional samples, as p directly influences the variance of the estimator. We discuss such "good" choices in section 4.3.3.

Variance

The variance of the Monte Carlo estimator F_N with N samples depends on the variance of the single-sample estimator F_1 (or, in other words, on the variance of the random variable f/p):

$$\begin{aligned} V(F_1) &= \int_\Omega \frac{f^2(x)}{p(x)} \,\mathrm{d}\mu(x) - I^2 \\ V(F_N) &= \frac{1}{N} V(F_1). \end{aligned} \tag{4.3.3}$$

In other words, the variance shrinks linearly with the number of samples [16], independently of the dimensionality of the problem.

This is what makes Monte Carlo integration suitable for solving high-dimensional integrals, since it does not suffer from the curse of dimensionality as opposed to other integration approaches, where the variance also depends on the dimensionality.

In addition to N, the variance also depends on a constant factor $V(F_1)$ that in turn depends on the PDF p from which the samples X_i are drawn. Good choices for p can in fact decrease the variance, as discussed in the next section.

4.3.3 Importance Sampling

The choice of the density p from which we draw samples for Monte Carlo integration has a direct impact on the variance of the estimator. Ideally, we would like to use a density directly proportional to the integrand itself, $p^* = f/I$, since in this case the variance would simplify to

$$\begin{aligned} V(F_N) &= \frac{1}{N}\left(\int_\Omega \frac{f^2(x)}{p^*(x)}\,\mathrm{d}\mu(x) - I^2\right) \\ &= \frac{1}{N}\left(\int_\Omega \frac{f^2(x)}{f(x)/I}\,\mathrm{d}\mu(x) - I^2\right) \\ &= \frac{1}{N}\left(\int_\Omega I f(x)\,\mathrm{d}\mu(x) - I^2\right) = 0. \end{aligned}$$

In other words, with a density proportional to the integrand, the variance of the Monte Carlo estimator is 0, and we would get a perfect estimate $F_1 \equiv f/(f/I) = I$ even with a single sample ($N = 1$) from p^*.

Unfortunately, in order to compute a density p^* (i.e. a function integrating to one) from an arbitrary function f (a function integrating to I), we need to know the normalization factor, which happens to be the integral value I we wanted to compute in the first place. As we usually do not know the integral value in advance, this strategy, while theoretically perfect, is not practical.

Instead we can use a density that is at least roughly proportional to the integrand or some factors of the integrand. This strategy is called *importance sampling*. While not perfect, it often lowers the estimator's variance, resulting in a lower error with a fixed number of samples. With importance sampling, the variance still decreases linearly with the number of samples, albeit with a lower constant factor.

4.3.4 Multiple Importance Sampling

Sometimes, finding a single density that approximates most of the integrand well is hard. Instead, we might find several simpler densities, each approximating the integrand well in a subset of the integration domain, or maybe approximating some factors of it well.

For multiple importance sampling (MIS) [17] we start with M sampling strategies with PDFs $p_1, \ldots, p_M$. The only restriction on the set of densities is that it must be possible to sample any value that contributes to the integral from at least one of the strategies, though not necessarily all of them:

$$f(x) \neq 0 \Rightarrow \exists i : p_i(x) \neq 0.$$

The multi-sample model. For multiple importance sampling with the multi-sample model, we draw n_i samples $X_{i1}, \ldots, X_{in_i}$ from each strategy $i = 1 \ldots M$, and combine all samples in the MIS estimator

$$F_{mis} = \sum_{i=1}^{M} \frac{1}{n_i} \sum_{j=1}^{n_i} w_i(X_{ij}) \frac{f(X_{ij})}{p_i(X_{ij})} \tag{4.3.4}$$

by weighting them with a set of weighting functions w_i, one per strategy. This estimator is unbiased as long as the weighting functions sum up to one for any given sample X, and only assign weight to strategies that potentially could have sampled X:

$$f(X) \neq 0 \Rightarrow \sum_{i=1}^{N} w_i(X) = 1$$
$$p_i(X) = 0 \Rightarrow w_i(X) = 0.$$

The one-sample model. For multiple importance sampling with the one-sample model, we first pick one of the strategies with probability c_i, and then draw one sample from this strategy. This results in the estimator

$$F = \frac{w_i(X) f(X)}{c_i p_i(X)}$$

with the same conditions for the weighting functions as above.

The balance heuristic. In either case, it turns out that simply averaging the samples from all strategies together, i.e. with weights such as

$$w_i \equiv \frac{n_i}{\sum_{j=1}^{M} N_j},$$

makes the estimator inherit some of the variance and thus convergence rate of the strategy with the highest variance. Instead, Veach and Guibas [17] propose the *balance heuristic*

$$w_i(X) = \frac{p_i(X)}{\sum_{j=1}^{M} p_j(X)} \tag{4.3.5}$$

and show that it is optimal, in the sense that for any given integral we cannot find "much better" weights, at least as long as the weights are restricted to be non-negative, and all sampling strategies are unbiased.

In recent works on MIS, Kondapaneni et al. [18] showed that variance can be reduced even further if negative weights are allowed, and West et al. [19] discuss MIS in the presence of a continuum of sampling strategies instead of a discrete set $p_1, \ldots, p_M$.

Others apply MIS in the presence of biased sampling strategies: Kaplanyan et al. [20] resort to a maximum distance heuristic instead, assigning weight one to the estimator with the lowest bias and zero to all others, while vertex connection and merging [21] [22] faces the additional challenge of transforming the biased and unbiased strategies to the same measure in order to enable MIS with the balance heuristic in the first place. We discuss how we handled such cases in section 6.3.5 for spectral regularization, and sections 7.7 and 7.8 for spatial regularization.

4.4 Photorealistic Image Synthesis: Theory

Photorealistic image synthesis is based on the physical properties of light, more specifically its radiometric quantities (section 4.4.1) and how they are affected by emission (section 4.4.2) and scattering at surfaces (section 4.4.3) or inside participating media (section 4.4.4), before being measured by a pixel sensor in a camera (section 4.4.5).

The rendering equation (section 4.4.6) summarizes these effects by describing the light that leaves some surface as the sum of emitted light and incident, scattered light. This equation is recursive, since the incident light at some point can be expressed as the exitant light at a previous point. In section 4.4.7 we expand this to a non-recursive form more suitable for the numerical solutions presented in section 4.5. This results in the concept of a light transport path as a sequence of points in space along which light can travel in straight lines from a light source to the camera. We classify different types of such paths in section 4.4.8, which will later help us discuss why some of them may cause issues with common numerical solutions.

4.4.1 Radiometry

Radiometric quantities describe physical properties of light, from the properties of a single photon all the way to how bundles of photons move through space. This section covers light moving through empty space; interactions with surfaces (section 4.4.3) and particles in a medium (section 4.4.4) are discussed later.

We only consider geometric optics with light moving along straight lines, and ignore wave-optical effects such as interference. We do however account for the wavelength of light in the sense that it affects what fraction of light is absorbed or reflected by a surface, and how strongly a light ray changes direction when entering another medium, e.g. when transitioning from air to glass.

We start out with a single photon. For our purposes, a photon is fully described by three properties:

$\mathbf{x}$, its position in 3D space in $[m^3]$,

ω, the direction in which it moves as a unitless, normalized 3D vector, and

λ, its wavelength in $[nm]$.

While lengths are typically measured in $[m]$, we make an exception for a photon's wavelength and measure it in $[nm]$. This helps distinguish the wavelength domain from other length-based domains such as the surface area measured in $[m^2]$, or distance traveled along a light ray in $[m]$. Furthermore, most applications of photorealistic image synthesis only simulate visible light, where the wavelength is roughly in the $[400, 750]$ nm range (see figure 4.6), allowing us to represent typical values without exponents or multiple decimal places.

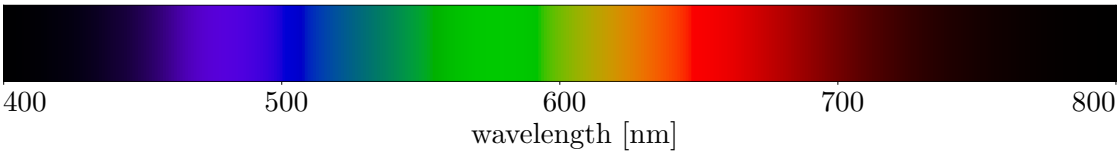


Figure 4.6: Visible light is roughly in the [400, 750] nm range.

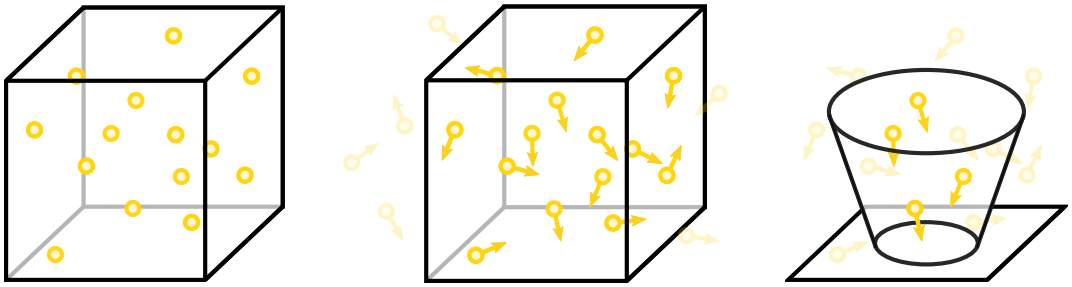


Figure 4.7: Radiant energy (**left**), radiant power (**center**) and radiance (**right**) quantify the energy of light in a piece of volume, per time, and per area and direction.

Individual photons. A photon within a medium moves with speed

$$c_m = \frac{c}{\eta_m} \ [m/s],$$

where $c \approx 3 \cdot 10^8 m/s$ is the speed of light in a vacuum, and $\eta_m > 1$ is the medium's refractive index.

The photon's energy

$$E_p = \frac{h \cdot c_m}{\lambda} \ [J] \tag{4.4.1}$$

depends on both its speed c_m and its wavelength λ, and is scaled by Planck's constant $h \approx 6.6 \cdot 10^{-34} Js$.

Since the typical energy of a photon in air with a wavelength of, say, 550nm (green) is still only $\approx 3.6 \cdot 10^{-19} J$, a single photon by itself is far from enough to light an entire room and hardly changes the sensor response of a typical camera. So instead, we now move on to regarding bundles of photons.

A collection of N photons with wavelength λ within some volume (figure 4.7 left) has *radiant energy*

$$Q = N \cdot E_p = N \cdot \frac{h \cdot c_m}{\lambda} \ [J].$$

The number of photons in a typical scene is large and thus the influence of a single photon on Q (and quantities depending on Q) is negligible, so for the rest of the thesis we ignore that these quantities actually occur in discrete, per-photon steps and treat them as continuous quantities instead.

Radiant power. Since photons are not stationary but move inside each medium with a speed c_m (figure 4.7 center), we can express the radiant energy within a volume over time as the *radiant power* or *flux*

$$\Phi = \frac{\mathrm{d}Q}{\mathrm{d}t} \ [J/s = W],$$

by counting the photons within that volume over time and accumulating their energy, independent of the direction in which they move and where they enter or leave the volume.

Radiance. Finally, as the end goal of photorealistic image synthesis is to measure the photons arriving from certain directions at a virtual camera sensor's surface, we restrict the radiant energy to photons passing through a certain area with a certain direction (figure 4.7 right) to get *radiance*

$$L(\mathbf{x}, \omega) = \frac{\mathrm{d}\Phi}{\mathrm{d}A(\mathbf{x})\,\mathrm{d}\sigma(\omega)} \ [W/(m^2 \cdot sr)]$$

to describe the differential power $\mathrm{d}\Phi$ per area $\mathrm{d}A(\mathbf{x})$ and solid angle $\mathrm{d}\sigma(\omega)$.

This quantity is the basis for photorealistic rendering. For instance, the energy measured by a virtual pixel sensor is defined as the incident radiance integrated over the sensor's area and acceptable incident directions, weighted by the sensor's sensitivity per location on the sensor and incident direction. However, in order to compute the incident radiance at a pixel, we have to consider the journey of a photon, starting at some light source, potentially interacting with (i.e. bounce off or get absorbed by) surfaces in a scene, before finally being detected by a pixel sensor.

The next four sections cover this journey and how radiance propagates through a scene. We start by discussing typical emitters, the physical principles behind them and how to describe their emission profiles in the context of radiant energy in section 4.4.2. Section 4.4.3 provides a closer look at how photons can interact with and scatter off surfaces along their path to the camera, and how to characterize the reflective properties of a surface. In addition, photons can interact with participating media between surfaces (section 4.4.4), although we keep this part short as it plays no major role in the contributions of this thesis. In the end, a pixel sensor (section 4.4.5) measures incident photons to form an image.

Finally, we discuss how to solve the corresponding equations for the pixel measurement in section 4.5, how the corresponding equations change when considering wavelength-dependent behavior of light in section 4.6 and what modifications are necessary to include fluorescence in section 4.7.

4.4.2 Light Sources

Light emission can be caused by a variety of physical processes. Incandescence is the emission of visible light due to high temperature, such as by the sun, fire, or old-fashioned light bulbs with a heated wire. Luminescence covers everything else, with electroluminescence being the most common one in artificial light sources (e.g. LEDs).

Meanwhile, photoluminescence refers to the emission of light that was previously absorbed by some substance, in particular fluorescence (see section 4.7) and phosphorescence. For accurate photorealistic image synthesis though we do not treat fluorescent and phosphorescent materials as light sources, as they are unable to illuminate a scene on their own, and instead simulate how they interact with incident light from other sources.

Ultimately, in the context of rendering we do not care which physical processes cause emission, we only care about the spectral, directional and spatial emission profiles. In this section we discuss how to mathematically model light sources in virtual scenes. Some of these models have direct physical counterparts, such as area light sources (a solid object whose surface emits photons, e.g. light bulbs or a hot stove top) and volumetric emitters (a gas that emits photons, e.g. fire, explosions or gas inside a fluorescent tube). Others do not exist in the real world but are useful to approximate certain effects.

Point lights

Point lights emit light from a single point in space, often evenly in all directions. Such light sources are physically impossible – even a spark from a bonfire occupies a small, finite volume in space – In the context of most rendering algorithms they represent a singularity. Depending on the overall scene configuration, this can be hard to deal with, as we explore in sections 4.5.1 and 4.8. In other cases they simply offer a useful approximation of tiny light sources without actually having to create scene geometry, or may even simplify some aspects, e.g. when computing direct illumination.

A point light is characterized by its position $\mathbf{p}$ and the power Φ in Watt emitted by the point light. This power is emitted continuously in directions ω with *radiant intensity*

$$I(\omega) = \frac{\mathrm{d}\Phi}{\mathrm{d}\sigma(\omega)} \ [W/sr]. \tag{4.4.2}$$

Area lights

Area lights emit light from a surface. The emitted radiance can vary both over the surface and over directions. The emitted radiance is the differential emitted power at surface point $\mathbf{x}$ in direction ω:

$$L_e(\mathbf{x}, \omega) = \frac{\mathrm{d}^2\Phi}{\mathrm{d}A(\mathbf{x}) \cos\theta \, \mathrm{d}\sigma(\omega)} \ [W/(m^2 \cdot sr)],$$

where θ is the angle between ω and the surface normal at point $\mathbf{x}$, and ω points in the direction along which the photon is emitted.

The area lights in this thesis either have a Lambertian emission profile

$$L_e(\mathbf{x}, \omega) \propto \cos\theta$$

or the more focused profile of a cosine-power emitter

$$L_e(\mathbf{x}, \omega) \propto \cos^k \theta$$

that grows sharper with increasing k.

Directional light sources

Taking the profile of a cosine-power emitter to the limit for $k \to \infty$ leads to a directional area light that exclusively emits in one direction. Such a directional light source is characterized by the emission direction ω and emitted power Φ, and we describe the emitted *radiant exitance* as the differential power Φ per surface area

$$E(\mathbf{x}) = \frac{\mathrm{d}\Phi}{\mathrm{d}A(\mathbf{x})} \ [W/m^2].$$

Directional light sources can also be used to efficiently approximate the effect of near-directional illuminants, such as the sun observed from a human-scale region on the earth's surface. In this case we no longer deal with emitting geometry and simply assume constant incident illumination from direction ω all over the scene in $[W/m^2]$. Like point lights, perfect directional emitters do not exist in the real world and are singularities that may challenge certain rendering algorithms.

Environment Maps

Environment maps are motivated by scenes where the illuminating environment is relatively far away from the rest of the scene, such as the sky, or a big room around a small object. In such cases varying positions within the scene hardly changes the incident illumination per direction. This allows us to skip some steps of simulating photon paths from their original emission source across some far-away objects, and only use the accumulated incident radiance at the local scene. As such, environment maps not only include light from actual light sources but also light that has previously interacted with other far away materials, e.g. clouds in the sky. Like point lights, an environment map's emission profile is described as differential power per solid angle in $[W/sr]$, although conceptually a point light's energy travels outward from a point, while an environment map emits energy inward, from a huge sphere surrounding the scene.

Volumetric emitters

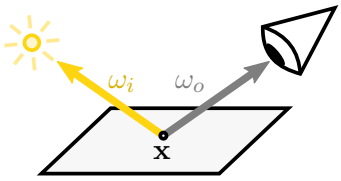
Volumetric emitters emit light from a three-dimensional volume instead of a two-dimensional surface. Since our contributions are orthogonal to volumetric rendering, we discuss volumetric light sources and other volumetric effects only briefly in section 4.4.4.

4.4.3 BSDF

The *bidirectional scattering distribution function*, BSDF for short, describes how light scatters at surfaces. It is defined as the ratio of differential exitant radiance in direction ω_o to differential irradiance $\mathrm{d}E$ from direction ω_i at a surface point $\mathbf{x}$:

$$f_r(\omega_i, \mathbf{x}, \omega_o) = \frac{\mathrm{d}L(\mathbf{x}, \omega_o)}{\mathrm{d}E(\mathbf{x}, \omega_i)} = \frac{\mathrm{d}L(\mathbf{x}, \omega_o)}{L(\mathbf{x}, \omega_i) \cos\theta_i \, \mathrm{d}\sigma(\omega_i)}. \tag{4.4.3}$$

We use the convention that both ω_i and ω_o point away from the surface. That is, ω_i points in the direction from which light arrives at $\mathbf{x}$, and ω_o points in the direction in which light leaves $\mathbf{x}$, as in the figure on the right. θ_i is the angle between ω_i and the surface normal.



As long as we ignore effects where incident light physically changes a surface (e.g. photobleaching or such high energy that the surface melts), this ratio is constant - if we double the overall incident irradiance, the exitant radiance into any direction doubles as well.

For more complex effects we need to introduce additional parameters to the BSDF. For instance, spectral rendering recognizes that this ratio may vary depending on the light's wavelength λ, resulting in a spectral BSDF $f_r(\omega_i, \mathbf{x}, \lambda, \omega_o)$. Fluorescent BSDFs $f_f(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o)$ may change the wavelength λ_i of the incident light to another wavelength λ_o during scattering. Subsurface scattering BSDF $f_{ss}(\omega_i, \mathbf{x}_i, \mathbf{x}_o, \omega_o)$ may transport light within the surface from an entry point $\mathbf{x}_i$ to an exit point $\mathbf{x}_o$. We often ignore parameters if the BSDF in question does not depend on them, e.g. if a BSDF is constant across a surface we may only write $f_r(\omega_i, \omega_o)$.

Physically plausible BSDFs. Physically plausible BSDFs are

- non-negative: $f_r \geq 0$,
- Helmholtz-reciprocal: $f_r(\omega_i, \mathbf{x}, \omega_o) = f_r(\omega_o, \mathbf{x}, \omega_i)$, and
- energy conserving: $\forall \omega_i : \int_{\mathbb{S}^2} f_r(\omega_i, \mathbf{x}, \omega_o) \cos\theta_o \, \mathrm{d}\sigma(\omega_o) \leq 1$.

  The exact formula for this requirement may change when introducing additional parameters (e.g. for fluorescence or subsurface scattering), but the basic idea remains that the surface may never scatter more than 100% of the incident energy from any incident direction ω_i, position x_i or wavelength λ_i.

These properties are necessary but not sufficient to yield a BSDF that could reasonably appear in the real world. For example, since these criteria do not restrict shape of the BSDF in any domain – spectral, directional or spatial – they allow for quite unnatural behavior, such as technically impossible discontinuities in the reflectance spectrum.

We investigate further restrictions on the spectral domain in chapter 5 where we aim to model reflectance and fluorescence spectra that better match real-world observations. On the other hand, we make use of technically unrealistic but physically plausible BSDFs in chapters 6 and 7 to approximate and converge to target BSDFs that are more realistic but also harder to simulate.

BSDF types. The following characteristics help us classify some aspects of the BSDFs that we use throughout this thesis:

- scattering hemisphere: The terms BRDF, BTDF and BSDF refer to the bidirectional reflectance, transmittance and scattering distribution function for surfaces that scatter in the upper, lower or both hemispheres with respect to the surface's normal. Unless we specifically care about surfaces that only reflect or transmit, we will use the term BSDF.

- analytical vs. data-driven: We can either use an analytical expressionas a BSDF, or measure the reflectance of a real-world material sample and use the measured data to reproduce the reflectance behavior in a virtual scene.

- physics based vs. phenomenological: We can further distinguish analytical expressions as either being derived from physical observations or models (e.g. microfacet models, conductors and dielectrics, see below) or lacking any physical motivation except that the resulting surface happens to look similar to existing surfaces.

- continuous (diffuse or glossy) vs. specular: We use the term diffuse for materials that scatter evenly in all directions. Glossy materials scatter more in some directions than others. Specular materials such as glass or mirrors scatter any incident photon into a single deterministic direction. In this case the exiting radiance is not actually differentiable with respect to incident direction, which means that the definition (equation 4.4.3) for the BSDF does not technically apply here. A common approach in practice is to use some sort of Dirac symbol for this case, along the lines of $f(\omega_i, \omega_o) = \delta_{\omega_o}(\omega_{ir})$, to denote that the surface scatters exclusively from ω_i toward the specular direction ω_{ir}. Strictly speaking, this expression is no longer a function, and we discuss the implications of this in section 4.8.

- wavelength-preserving (elastic) vs. fluorescent (inelastic): Many real-world materials, and most current rendering systems, do not change the wavelength of a photon in the scattering process. Fluorescent surfaces and volumes, however, do just that, adding another layer of complexity to the simulation, as discussed in section 4.7.

The remainder of this section presents the surface types and BSDF models that we used to render the images later in this thesis.

Diffuse Lambertian BRDFs scatter incident light evenly over the upper hemisphere:

$$f_r(\omega_i, \mathbf{x}, \omega_o) = \frac{1}{\pi} a.$$

Such a surface appears equally bright with albedo a from every direction. If $a \in [0, 1]$ the BRDF is energy conserving.

Mirrors reflect (a fraction a of) the incident light from direction ω_i toward a single specular direction

$$\omega_{ir} = 2\langle \mathbf{n}, \omega_i \rangle \mathbf{n} - \omega_i$$

depending on the surface normal $\mathbf{n}$, resulting in a reflected radiance of

$$L_o(\mathbf{x}, \omega_{ir}) = a \cdot L_i(\mathbf{x}, \omega_i).$$

In the context of the BRDF definition in equation 4.4.3, L_o only depends on $L_i(\omega_i)$ and does not change at all for varying differential irradiance from any other incident direction, resulting in a discontinuity at ω_i and making L_o non-differentiable at ω_i. In other words, the derivative of L_o w.r.t. differential irradiance is 0 everywhere except at ω_i where it does not exist. Hence there exists no function in the classical sense that we can use as a BRDF f_r to describe such a surface. We discuss how to handle such specular cases in light transport simulations in section 4.8.

Dielectrics both reflect and transmit light. Just like mirrors they are perfectly specular, meaning they only scatter in a discrete set of directions, and have no corresponding BSDF since the outgoing radiance is non-differentiable for the specular directions. The transmitted direction ω_{it} lies in the plane of the incident direction ω_i and the surface normal, and its angle θ_t to the normal follows Snell's law

$$\sin\theta_t = \sin\theta_i \cdot \frac{n_i}{n_t},$$

where n_i and n_t are the refractive indices of the substances on the incident and exitant side of the surface. The refractive indices usually depend on the wavelength, so transmitted light with different wavelengths ends up in different directions.

The Fresnel term F describes how much of the incident radiance from ω_i gets reflected:

$$\begin{aligned} L_o(\mathbf{x}, \omega_{ir}) &= L_i(\mathbf{x}, \omega_i) F(\omega_i) \\ L_o(\mathbf{x}, \omega_{it}) &= L_i(\mathbf{x}, \omega_i)(1 - F(\omega_i)). \end{aligned}$$

For non-polarized light we can compute F as

$$F(\omega_i) = 0.5 \cdot \left(\left| \frac{n_i \cos\theta_i - n_t \cos\theta_t}{n_i \cos\theta_i + n_t \cos\theta_t} \right|^2 + \left| \frac{n_i \cos\theta_t - n_t \cos\theta_i}{n_i \cos\theta_t + n_t \cos\theta_i} \right|^2 \right).$$

Conductors are mirrors where the albedo is governed by the Fresnel equation. Similar to dielectrics, their reflective albedo for a specific incident direction is described by the Fresnel term, but they absorb any light transmitted into the material:

$$L_o(\mathbf{x}, \omega_{ir}) = F(\omega_i) L_i(\mathbf{x}, \omega_i).$$

They have a wavelength dependent complex refractive index $n = \eta + i\kappa$, giving them their typical metal-like colors.

Microfacet models assume that a surface consists of many tiny microfacets with a simple micro-BSDF, and whose orientation around the geometric "global" surface normal is described statistically by a normal distribution function (NDF) as in figure 4.8.

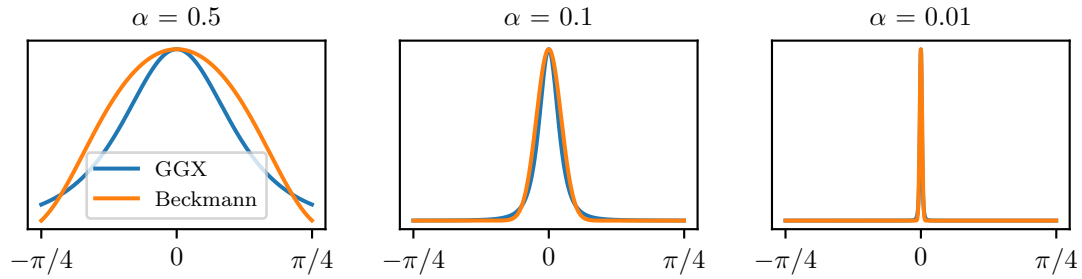


Figure 4.8: The GGX and Beckmann normal distribution function plotted over the angle between microfacet normal and geometric normal, with shrinking roughness α.

For instance, the Torrance-Sparrow model uses perfectly specular mirror facets, resulting in the BRDF

$$f_r(\omega_i, \mathbf{x}, \omega_o) = \frac{D_\alpha(\omega_h)F(\omega_i, \omega_h)G(\omega_o, \omega_i, \omega_h)}{4\langle \mathbf{n}, \omega_o\rangle\langle \mathbf{n}, \omega_i\rangle}.$$

The surface appears rougher if the microfacet normals deviate more from the overall surface normal $\mathbf{n}$. As such, the resulting BSDF is often termed "rough conductor" or "rough dielectric", if the microfacets are conductors or dielectrics. In common NDFs such as the GGX or Beckmann distribution [23][24, ch. 8.4.2] this roughness can be controlled by a user-parameter $\alpha \in (0, 1]$ to stretch or squash the normal distribution function, as illustrated by figure 4.8. For $\alpha \to 0$ both the GGX and Beckmann NDF converge to a Dirac peak, with all of its mass concentrated at $\omega_h = \mathbf{n}$ and evaluating to 0 everywhere else. This means that all microfacet normals align perfectly with the surface normal, and the appearance degenerates to that of the micro-BSDF itself, usually a perfect mirror or a perfect dielectric.

The cosines in the Fresnel term F are taken with respect to the normal ω_h of the microfacet that actually scatters light from ω_i to ω_o, not with respect to the global surface normal $\mathbf{n}$. Thanks to the deterministic nature of specular reflection and dielectric transmission, this so-called half vector ω_h is unique for both mirror facets and perfectly specular dielectric facets.

The shadowing masking function G describes how the microfacets on the surface occlude each other. This includes both facets blocking incident light from arriving at nearby facets, as well as blocking light from leaving nearby facets. Technically, the latter case results in multiple scattering, where light is reflected off more than one microfacet before eventually leaving the surface. For simplicity, we ignore this and only use single scattering, resulting in a small energy loss for rough surfaces, particularly at grazing angles.

Layered materials combine multiple thin layers of semi-transparent substances, with either an opaque or transmissive base [25]. Such BSDFs can achieve a wide variety of appearances, and may result in expressions that contain both specular and continuous components.

4.4.4 Participating Media

Participating media can affect the light field in a scene by emitting, absorbing and scattering light within the 3D volume between surfaces. The emission coefficient μ_e, scattering coefficient μ_s and absorption coefficient μ_a describe the likelihood of these events. The phase function ρ describes the directional distribution of scattered directions whenever a light ray interacts with a scattering particle.

As opposed to surfaces, where we only dealt with radiance changes in a single point, we are now interested in the radiance changes along light rays. Thus we further combine the scattering and absorption coefficients into the extinction coefficient $\mu_t = \mu_s + \mu_a$, since we lose energy along a particular ray both due to absorption and to out-scattering in other directions. The fraction of energy that remains after absorption and out-scattering, after a light ray travels some distance t in direction ω starting from y, is the *transmittance*

$$T(\mathbf{y}, \omega, t) = e^{-\int_0^t \mu_t(\mathbf{y}+u\omega)\,\mathrm{d}u}.$$

On the other hand, we gain energy along the ray both due to emitted radiance L_e and due to in-scattered radiance L_s from other directions, which in turn decreases according to the transmittance. This is captured by the radiative transfer equation

$$L(\mathbf{x}, \omega) = \int_0^d T(\mathbf{y}, \omega, t) \left(\mu_e(\mathbf{y}) L_e(\mathbf{y}, \omega) + \mu_s(\mathbf{y}) L_s(\mathbf{y}, \omega)\right) \mathrm{d}t,$$

integrating all points $\mathbf{y} = \mathbf{x} - t\omega$ along the ray arriving at $\mathbf{x}$. In practice, d is either infinite if the medium extends forever, or limited by either the extent of the medium or the distance to the next surface. For more detail we refer to [24, ch. 11 and 15] and [26].

4.4.5 Pixel Sensors and Camera Models

Pixel sensors measure the incident light field on an image plane. The sensor sensitivity function $W^j(\mathbf{x}, \omega)$ describes how a particular pixel j reacts to incident radiance L_i at position $\mathbf{x}$ from direction ω. We can thus compute the total measurement I^j at a pixel j by integrating over the sensor surface S and all incident directions in the upper hemisphere $\mathbb{H}^2$ above the image plane:

$$I^j = \int_{\mathcal{S}} \int_{\mathbb{H}^2} W^j(\mathbf{x}, \omega) L_i(\mathbf{x}, \omega) \cos\theta \,\mathrm{d}\sigma(\omega)\,\mathrm{d}A(\mathbf{x}).$$

Again, θ is the angle between ω and the surface normal $\mathbf{n}$ at $\mathbf{x}$. In cases like this where we care about the cosine and not the angle itself, we sometimes write the dot product $\langle \omega, \mathbf{n} \rangle$ instead of $\cos\theta$. Solving this integral is the final goal of photorealistic image synthesis.

Usually the image plane does not simply float freely in space, but is at the end of some sort of camera system. The simplest example is a pinhole camera, where all light rays have to pass through a single point – the pinhole – to reach the pixel sensor. This is easy to simulate and results in perfectly sharp images, but cannot mimic the appearance arising from modern cameras, where one or multiple lenses and/or mirrors in front of the sensors cause effects such as depth of field, distortions, vignetting or chromatic aberration. We use the thin lens camera model [24, ch. 6.2.3] with a single lens in front of the image plane to at least achieve some basic depth of field.

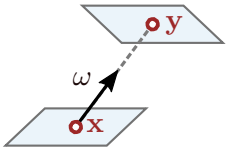
4.4.6 Rendering Equation

In the previous section we introduced the equation for the measurement I^j at a pixel j:

$$I^j = \int_{\mathcal{S}} \int_{\mathbb{H}^2} W^j(\mathbf{x}, \omega) L_i(\mathbf{x}, \omega) \cos\theta \, \mathrm{d}\sigma(\omega) \, \mathrm{d}A(\mathbf{x}). \tag{4.4.4}$$

Ultimately, this is what we need to solve for each pixel on the image plane in order to create a photorealistic image.

The first step toward this solution is observing that the incident radiance $L_i(\mathbf{x}, \omega)$ at some point $\mathbf{x}$ is equal[5] to the exitant radiance at the first visible surface vertex $\mathbf{y}$ when tracing a ray from $\mathbf{x}$ in direction ω:



$$I^j = \int_{\mathcal{S}} \int_{\mathbb{H}^2} W^j(\mathbf{x}, \omega) L_o(\mathbf{y}, -\omega) \, \mathrm{d}\sigma(\omega) \, \mathrm{d}A(\mathbf{x}).$$

The rendering equation [27] tells us how to compute this outgoing radiance as the sum of emitted and scattered radiance at $\mathbf{y}$:

$$L_o(\mathbf{y}, \omega_o) = L_e(\mathbf{y}, \omega_o) + \int_{\mathbb{S}^2} L_i(\mathbf{y}, \omega_i) f_r(\omega_i, \mathbf{y}, \omega_o) |\cos\theta_i| \, \mathrm{d}\sigma(\omega_i). \tag{4.4.5}$$

To obtain the scattered radiance we collect the incident radiance L_i scaled by the BSDF f_r from all directions on the unit sphere $\mathbb{S}^2$. The cosine $|\cos\theta_i| = |\langle \mathbf{n}_y, \omega_i \rangle|$ between the surface normal $\mathbf{n}_y$ at $\mathbf{y}$ and the incident direction ω_i transforms the incident radiance from per solid angle around $\mathbf{y}$ to per projected solid angle at the surface around $\mathbf{y}$. This is necessary since the BSDF describes the distribution of outgoing radiance given the incident differential radiance *at the surface*, but we integrate incident radiance with the solid angle measure for directions on the unit sphere.

For the remainder of section 4.4 we will use the notation $\langle \mathbf{n}, \omega \rangle$ to denote the absolute value of the cosine between a normal $\mathbf{n}$ and direction ω, as not all angles will be measured with respect to the normal $\mathbf{n_y}$ at $\mathbf{y}$.

Without attenuation by participating media the incident radiance $L_i(\mathbf{y}, \omega_i)$ is equal to the exitant radiance $L_o(\mathbf{z}, -\omega_i)$ at the first visible surface vertex $\mathbf{z}$ when tracing a ray from $\mathbf{y}$ in direction ω_i. This outgoing radiance can in turn be computed by evaluating rendering equation for $L_o(\mathbf{z}, -\omega_i)$. So while the rendering equation neatly captures what happens at a single surface point, the 2D integral actually hides a recursive, infinite-dimensional integration problem.

[5] This is no longer true if participating media attenuate the light along the path from $\mathbf{y}$ to $\mathbf{x}$. Section 4.4.4 discusses how to handle this.

4.4.7 Path Space

This section briefly shows how to transform the compact but recursive form of the rendering equation to an infinite but non-recursive expression that is more convenient for deriving the numerical solutions in section 4.5. We do this in two steps:

1. Change the integration domain from incident directions to scene surfaces
2. Iteratively insert the equation in itself to resolve the recursion

Following equation 4.1.5, we can transform an integral from solid angle measure to surface area measure as

$$\int_{\mathbb{S}^2} g(\omega, \mathbf{y}) \langle \mathbf{n_y}, \omega \rangle \, \mathrm{d}\sigma(\omega) = \int_{\mathcal{M}} g(\mathbf{y} \to \mathbf{z}, \mathbf{y}) \langle \mathbf{n_y}, \mathbf{y} \to \mathbf{z} \rangle V(\mathbf{y}, \mathbf{z}) \frac{\langle \mathbf{n_z}, \mathbf{z} \to \mathbf{y} \rangle}{||\mathbf{y} - \mathbf{z}||^2} \, \mathrm{d}A(\mathbf{z}),$$

where

- $g(\mathbf{y} \to \mathbf{z}, \mathbf{y}) = g(\frac{\mathbf{z}-\mathbf{y}}{||\mathbf{z}-\mathbf{y}||}, \mathbf{y})$ is any function for $\mathbf{y}$ and the normalized direction pointing from $\mathbf{y}$ to $\mathbf{z}$,
- $\langle \mathbf{n_z}, \mathbf{z} \to \mathbf{y} \rangle$ is the absolute value of the cosine between the surface normal $\mathbf{n_z}$ at $\mathbf{z}$ and the direction $\mathbf{z} \to \mathbf{y}$,
- the visibility term $V(\mathbf{z}, \mathbf{y})$ is either 1 if $\mathbf{z}$ is the closest surface to $\mathbf{y}$ in direction $\mathbf{y} \to \mathbf{z}$, or 0 if there is another surface between $\mathbf{y}$ and $\mathbf{z}$.

We bundle the cosine and distance factors into the *geometry term*

$$G(\mathbf{y}, \mathbf{z}) := \frac{\langle \mathbf{n_y}, \mathbf{y} \to \mathbf{z} \rangle \cdot \langle \mathbf{n_z}, \mathbf{z} \to \mathbf{y} \rangle}{||\mathbf{y} - \mathbf{z}||^2}.$$

In addition, we express the integrand $L_i \cdot f_r \cdot \cos$ depending on vertices instead of directions, with arrows representing the direction of the light flow:

- $L_i(\mathbf{y} \leftarrow \mathbf{x}) := L_i(\mathbf{y}, \frac{\mathbf{x}-\mathbf{y}}{||\mathbf{x}-\mathbf{y}||})$ for radiance arriving at $\mathbf{y}$ from $\mathbf{x}$,
- $L_o(\mathbf{y} \to \mathbf{z}) := L_o(\mathbf{y}, \frac{\mathbf{z}-\mathbf{y}}{||\mathbf{z}-\mathbf{y}||})$ for radiance leaving $\mathbf{y}$ toward $\mathbf{z}$, and
- $f_r(\mathbf{z} \to \mathbf{y} \to \mathbf{x}) := f_r(\frac{\mathbf{z}-\mathbf{y}}{||\mathbf{z}-\mathbf{y}||}, \mathbf{y}, \frac{\mathbf{x}-\mathbf{y}}{||\mathbf{x}-\mathbf{y}||})$ for the BSDF at $\mathbf{y}$, and sometimes replace the arrows with commas for brevity if the context is clear (i.e. $f_r(\mathbf{z}, \mathbf{y}, \mathbf{x})$).

This allows us to rewrite the rendering equation as

$$\begin{aligned} L_o(\mathbf{x}_1 \to \mathbf{x}_0) = & L_e(\mathbf{x}_1 \to \mathbf{x}_0) + \\ & \int_{\mathbb{S}^2} L_i(\mathbf{x}_1 \leftarrow \mathbf{x}_2) f_r(\mathbf{x}_2 \to \mathbf{x}_1 \to \mathbf{x}_0) V(\mathbf{x}_1, \mathbf{x}_2) G(\mathbf{x}_1, \mathbf{x}_2) \, \mathrm{d}A(\mathbf{x}_2). \end{aligned}$$

Since the incident radiance at $\mathbf{x}_1$ from $\mathbf{x}_2$ equals[6] the exitant radiance at $\mathbf{x}_2$ toward $\mathbf{x}_1$, we can in turn apply the rendering equation to $L_i(\mathbf{x}_1 \leftarrow \mathbf{x}_2) = L_o(\mathbf{x}_2 \rightarrow \mathbf{x}_1)$ in the integrand:

$$\begin{aligned}
&\int_{\mathbb{S}^2} L_i(\mathbf{x}_1 \leftarrow \mathbf{x}_2) f_r(\mathbf{x}_2 \rightarrow \mathbf{x}_1 \rightarrow \mathbf{x}_0) V(\mathbf{x}_1, \mathbf{x}_2) G(\mathbf{x}_1, \mathbf{x}_2) \,\mathrm{d}A(\mathbf{x}_2) \\
&= \int_{\mathbb{S}^2} \left(L_e(\mathbf{x}_2 \rightarrow \mathbf{x}_1) + \int_{\mathcal{M}} L_i(\mathbf{x}_2 \leftarrow \mathbf{x}_3) f_r(\mathbf{x}_3 \rightarrow \mathbf{x}_2 \rightarrow \mathbf{x}_1) V(\mathbf{x}_2, \mathbf{x}_3) G(\mathbf{x}_2, \mathbf{x}_3) \,\mathrm{d}A(\mathbf{x}_3) \right) \\
&\qquad \cdot f_r(\mathbf{x}_2 \rightarrow \mathbf{x}_1 \rightarrow \mathbf{x}_0) V(\mathbf{x}_1, \mathbf{x}_2) G(\mathbf{x}_1, \mathbf{x}_2) \,\mathrm{d}A(\mathbf{x}_2).
\end{aligned}$$

Repeating this indefinitely allows us to rewrite $L_o(\mathbf{x}_1 \rightarrow \mathbf{x}_0)$ as an infinite sum of finite-dimensional integrals:

$$\begin{aligned}
L_o(\mathbf{x}_1 \rightarrow \mathbf{x}_0) = &L_e(\mathbf{x}_1 \rightarrow \mathbf{x}_0) + \\
&\sum_{k=2}^{\infty} \int_{\mathcal{M}^{k-1}} L_e(\mathbf{x}_k \rightarrow \mathbf{x}_{k-1}) \cdot \\
&\qquad \prod_{j=1}^{k-1} f_r(\mathbf{x}_{j+1}, \mathbf{x}_j, \mathbf{x}_{j-1}) V(\mathbf{x}_j, \mathbf{x}_{j+1}) G(\mathbf{x}_j, \mathbf{x}_{j+1}) \,\mathrm{d}A(\mathbf{x}_{j+1}).
\end{aligned}$$

In turn, this allows us to describe the pixel measurement without recursions as

$$\begin{aligned}
I^j = \sum_{k=1}^{\infty} \int_{\mathcal{M}^{k+1}} &W(\mathbf{x}_0 \leftarrow \mathbf{x}_1) V(\mathbf{x}_0, \mathbf{x}_1) G(\mathbf{x}_0, \mathbf{x}_1) \\
&\cdot \prod_{j=1}^{k-1} f_r(\mathbf{x}_{j+1}, \mathbf{x}_j, \mathbf{x}_{j-1}) V(\mathbf{x}_j, \mathbf{x}_{j+1}) G(\mathbf{x}_j, \mathbf{x}_{j+1}) \\
&\cdot L_e(\mathbf{x}_k \rightarrow \mathbf{x}_{k-1}) \,\mathrm{d}A(\mathbf{x}_k) \dots \mathrm{d}A(\mathbf{x}_0).
\end{aligned} \tag{4.4.6}$$

Finally, we abbreviate the integrand as the *measurement contribution function*

$$\begin{aligned}
f^j(\mathbf{x}_0, \dots, \mathbf{x}_k) = &W^j(\mathbf{x}_0, \mathbf{x}_1) V(\mathbf{x}_0, \mathbf{x}_1) G(\mathbf{x}_0, \mathbf{x}_1) \\
&\cdot \prod_{j=1}^{k-1} f_r(\mathbf{x}_{j+1} \rightarrow \mathbf{x}_j \rightarrow \mathbf{x}_{j-1}) V(\mathbf{x}_j, \mathbf{x}_{j+1}) G(\mathbf{x}_j, \mathbf{x}_{j+1}) \\
&\cdot L_e(\mathbf{x}_k \rightarrow \mathbf{x}_{k-1})
\end{aligned} \tag{4.4.7}$$

and switch from an infinite sum of finite-dimensional integrals to a single integral over the union of all the integration domains $\mathcal{P} := \bigcup_{j=2}^{\infty} \mathcal{M}^j$. This new integration domain is the *path space* $\mathcal{P}$. It contains paths as sequences of vertices $\mathbf{X} = (\mathbf{x}_0, \dots, \mathbf{x}_k)$. All paths with non-zero measurement contribution successfully connect a vertex $\mathbf{x}_0$ on the image plane to an emitting vertex $\mathbf{x}_k$, without occlusion between adjacent vertices $\mathbf{x}_j, \mathbf{x}_{j+1}$, and are thus valid *light transport paths*. Figure 4.9 shows an example. We integrate $\mathcal{P}$ using the combined product area measure $\mathrm{d}A((\mathbf{x}_0, \dots, \mathbf{x}_k)) = \mathrm{d}A(\mathbf{x}_k) \cdots \mathrm{d}A(\mathbf{x}_0)$, resulting in

$$I^j = \int_{\mathcal{P}} f^j(\mathbf{X}) \,\mathrm{d}A(\mathbf{X}).$$

[6] Again, only in the absence of participating media.

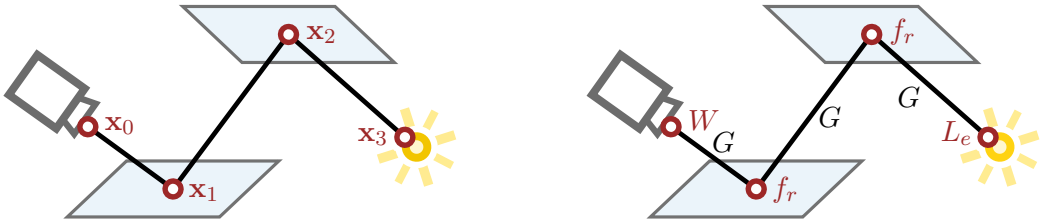


Figure 4.9: **Left**: A path $(\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ of length 4, with camera vertex $\mathbf{x}_0$, surface vertices $\mathbf{x}_1$ and $\mathbf{x}_2$, and light vertex $\mathbf{x}_3$. **Right**: Contributing factors to the path's contribution: The sensor sensitivity W, geometry terms G, BSDFs f_r and emitted light L_e.

4.4.8 Path Classification

This section introduces the Heckbert notation [28] for discussing particular types of light transport paths based on the properties of the path vertices. Vertices can be

- On the camera **E**
- On a light source **L**
- On a non-emitting surface between the camera and light source:
  - Specular vertices **S** with singular reflectance behavior (e.g. mirrors or glass)
  - Diffuse vertices **D** with a continuous BSDF

Some surfaces have both a specular and a diffuse component. In this case the vertex type is determined by the incident and outgoing direction of the particular path at that vertex. If the two directions align perfectly with a specular singularity, we call the vertex specular. Otherwise, we call it diffuse.

Usually we are not interested in the particular length of a path, and only care if a certain sequence of interactions occurs along the path. For this purpose we combine the vertex type symbols in regular expressions following [28], for example:

- ED^+L describes paths with only diffuse vertices between light and camera,
- $E(D|S)^*SL$ describes paths with at least one specular vertex right before the light, and
- $E(D|S)^*SDS(D|S)^*L$ describes paths with a specular - diffuse - specular interaction.

The classification can be extended depending on the context, e.g. to further distinguish spatially and directionally singular and continuous light sources, or to include other dimensions such as the spectral domain.

4.5 Photorealistic Image Synthesis: Solutions

This section presents common numerical solutions to compute the measurement I^j at a pixel j:

$$I^j = \int_{\mathcal{P}} f^j(\mathbf{X}) \, \mathrm{d}A(\mathbf{X}).$$

Solving this integral analytically is completely impractical due to the complexity of typical scenes. Instead, we resort to numerical integration for an approximate solution. Deterministic methods are ill-suited for this problem as they suffer from the curse of dimensionality: Their convergence speed depends not only on the number of function evaluations, but also on the dimensionality of the integrand. Instead, we use Monte Carlo integration (section 4.3), as its convergence rate is independent of the dimensionality, by drawing N paths from some PDF p on the path space $\mathcal{P}$ and combining them in the N-sample estimator F_N:

$$I^j \approx F_N := \frac{1}{N} \sum_{i=1}^{N} \frac{f^j(\mathbf{X}_i)}{p(\mathbf{X}_i)}.$$

The rendering algorithms in this section use different sampling strategies and thus different PDFs p, but all of them sample a path $\mathbf{X} = (\mathbf{x}_0, \ldots, \mathbf{x}_k)$ by building it vertex by vertex[7]. As a result, the corresponding PDFs can be separated, i.e.

$$p((\mathbf{x}_0, \ldots, \mathbf{x}_k)) = \prod_{i=0}^{k} p_A(\mathbf{x}_i),$$

into a product of surface-area-measure PDFs $p_A(\mathbf{x}_i)$. With some sampling strategies, $p_A(\mathbf{x}_i)$ may depend on previous or subsequent vertices, in which case $p_A(\mathbf{x}_i)$ stands for the conditional PDF, usually $p_A(\mathbf{x}_i) := p_A(\mathbf{x}_i|\mathbf{x}_{i+1})$ or $p_A(\mathbf{x}_i) := p_A(\mathbf{x}_i|\mathbf{x}_{i-1}, \mathbf{x}_{i-2})$.

4.5.1 Forward Path Tracing

Forward path tracing (PT) samples paths incrementally starting from the camera, as illustrated by figure 4.10. We first sample a vertex $\mathbf{x}_0$ on the image plane. Next we trace a ray from the camera towards a random direction and intersect it with the scene to get the first scene vertex $\mathbf{x}_1$. Then we keep sampling and appending vertices to the path until we eventually find a vertex $\mathbf{x}_k$ that emits light. Assuming the BSDF at $\mathbf{x}_k$ is zero we can then terminate the path, evaluate its measurement contribution (equation 4.4.7) and PDF, and add their ratio to the estimator.

This means that the PDF of the camera vertex $p_A(\mathbf{x}_0)$ does not depend on anything. The PDF of the first scene vertex $p_A(\mathbf{x}_1|\mathbf{x}_0)$ depends on the camera vertex, since we construct it by tracing a ray starting from $\mathbf{x}_0$ in a direction ω_1 sampled from the camera. We sample all subsequent vertices $\mathbf{x}_i$ by importance sampling the BSDF at $\mathbf{x}_i$ for the given previous direction ω_{i-1}, so the PDF of all subsequent vertices $p_A(\mathbf{x}_i|\mathbf{x}_{i-1}, \mathbf{x}_{i-2})$ depends on the two previous vertices.

[7]Not all rendering algorithms work like this. For instance, renderers based on Markov Chain Monte Carlo mutate an existing, complete path to generate a new path.

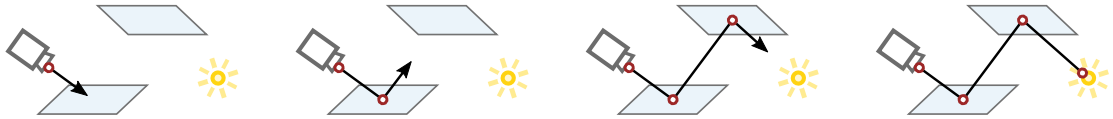
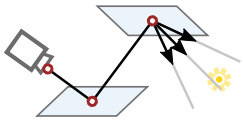


Figure 4.10: Path tracing constructs paths by sampling a point and direction on the camera (**left**), and then iteratively tracing rays until we find a light vertex (**right**).



Point lights. Path tracing cannot sample paths $(\mathbf{x}_0, \ldots, \mathbf{x}_k)$ with $\mathbf{x}_k$ on a point light. We discuss such cases in more detail in section 4.8, but the main problem is illustrated on the right: No direction ω_k from a continuous sampling decision based on the BSDF at $\mathbf{x}_{k-1}$ can ever align with a discrete, predetermined point light position – while it is possible to explicitly evaluate the PDF at $\mathbf{x}_{k-1}$ for the direction leading to the point light, it is theoretically impossible to sample that exact direction from a continuous PDF. Similarly, if the surface at $\mathbf{x}_{k-1}$ is singular (e.g. a mirror or glass), we still make a continuous sampling decision at the previous vertex $\mathbf{x}_{k-2}$, and specularly scattering that direction at $\mathbf{x}_{k-1}$ will miss the point light as well.

4.5.2 Next Event Estimation

In basic path tracing all importance sampling decisions are based solely on the local BSDF. This strategy completely neglects the influence of the incident radiance on the integrand, causing high variance that manifests as strong noise in the rendered image.

Generally, the incident radiance L_i at some vertex $\mathbf{x}_i$ consists of both direct illumination (i.e. light that reaches $\mathbf{x}_i$ straight from an emitter) and indirect illumination (i.e. light that was scattered at least once before reaching $\mathbf{x}_i$). While importance sampling indirect illumination remains a hard problem that is currently being addressed by data-driven methods [29], importance sampling the direct illumination is often quite practical.

To this end, *next event estimation* (NEE) connects a path prefix $(\mathbf{x}_0, \ldots, \mathbf{x}_i)$ to a randomly sampled vertex $\mathbf{x}_{nee}$ on an emitter to form a full path $(\mathbf{x}_0, \ldots, \mathbf{x}_i, \mathbf{x}_{nee})$ (see figure 4.11). If $\mathbf{x}_i$ and $\mathbf{x}_{nee}$ are not occluded, i.e. $V(\mathbf{x}_i, \mathbf{x}_{nee}) = 1$, and the BSDF at $\mathbf{x}_i$ evaluates to some non-zero value, i.e. $f_r(\mathbf{x}_{nee} \to \mathbf{x}_i \to \mathbf{x}_{i-1}) > 0$, this path is a valid light transport path and we can accumulate its contribution to the estimator. Technically, we also accumulate the contribution to the estimator if the path is not valid, although in this case it has no contribution due to the visibility term or BSDF being zero.

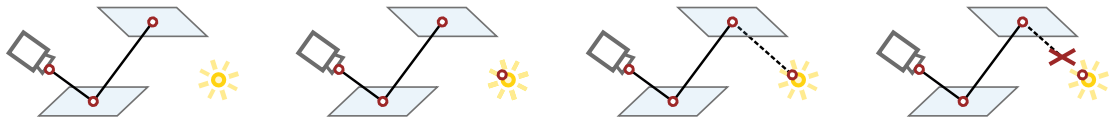
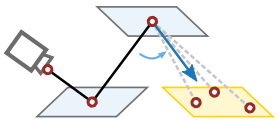


Figure 4.11: Next event estimation (NEE) starts with an incomplete path prefix (**left**), samples a vertex on a light source (**center left**) and connects the path prefix to the light vertex (**center right**) to form a complete light transport path. If the light source is occluded (**right**), the visibility term V evaluates to 0 and the path has no contribution.

Sampling a light vertex. In order to sample a light vertex, we first sample an emitter and then sample a position on that emitter. More precisely, we select an emitter based on its overall power, and a position on the emitter based on its emission profile, resulting in a PDF $p_A^{NEE}(\mathbf{x}_{nee})$ that does not depend on the path prefix. More elaborate approaches have the PDF $p_A(\mathbf{x}_{nee}|\mathbf{x}_i, \mathbf{x}_{i-1})$ depend on previous vertices in the scene [30, 31, 32] to favor emitters with a potentially higher contribution to the last surface vertex $\mathbf{x}_i$ based on their distance, visibility, BSDF value, direction (below/above the surface) or orientation (pointing toward or away from the surface).

If the emission profile does not change over the emitter, the second step corresponds to uniform sampling of the emitter's surface with PDF $p \equiv 1/A_{emitter}$, i.e. one over the emitter's surface area. For point lights or directional emitters we skip this, and for environment maps we importance sample a direction from the map instead.

Specular surfaces. Next event estimation cannot sample paths $(\mathbf{x}_0, \ldots, \mathbf{x}_k)$ where $\mathbf{x}_{k-1}$ is specular (e.g. a mirror or glass), as illustrated on the right. If $\mathbf{x}_{k-1}$ is specular, the previous vertex $\mathbf{x}_{k-2}$ determines the singular direction(s) (blue) from which the surface at $\mathbf{x}_{k-1}$ scatters light toward $\mathbf{x}_{k-2}$. This predetermined singular direction can never align with a continuously sampled vertex on an area light, continuously sampled direction on an environment map, or even a discretely sampled point light vertex. Any such NEE vertex $\mathbf{x}_k$ results in a direction $\mathbf{x}_{k-1} \to \mathbf{x}_k$ for which the surface at $\mathbf{x}_{k-1}$ scatters zero light toward $\mathbf{x}_{k-2}$. We discuss such singular paths in more detail in section 4.8.



Multiple Importance Sampling

NEE relies on path tracing to provide a path prefix and only samples the last vertex of a potential path. This means we still need to run path tracing in order to provide path prefixes for NEE, and thus have two strategies that potentially yield a valid light transport path: either we accidentally stumble across an emitting vertex after sampling a direction from the BSDF, or we deliberately connect to an emitting vertex through NEE (see figure 4.12). The two strategies have the following PDFs:

$$p_A^{PT}((\mathbf{x}_0, \ldots, \mathbf{x}_k)) = p_A(\mathbf{x}_0) p_A(\mathbf{x}_1|\mathbf{x}_0) \prod_{i=2}^{\mathbf{k}} p_A^{BSDF}(\mathbf{x}_i|\mathbf{x}_{i-1}, \mathbf{x}_{i-2})$$

$$p_A^{NEE}((\mathbf{x}_0, \ldots, \mathbf{x}_k)) = p_A(\mathbf{x}_0) p_A(\mathbf{x}_1|\mathbf{x}_0) \prod_{i=2}^{\mathbf{k-1}} p_A^{BSDF}(\mathbf{x}_i|\mathbf{x}_{i-1}, \mathbf{x}_{i-2}) \cdot p_A^{NEE}(\mathbf{x}_k).$$

We can combine the contributions of both cases using the multi-sample MIS model (equation 4.3.4). In particular, whenever we succeed in creating a path $\mathbf{X} = (\mathbf{x}_0, \ldots, \mathbf{x}_k)$ through either PT or NEE, its contribution to the estimator is either

$$w_{PT}(\mathbf{X}) \cdot \frac{f^j(\mathbf{X})}{p_A^{PT}(\mathbf{X})} \qquad \text{or} \qquad w_{NEE}(\mathbf{X}) \cdot \frac{f^j(\mathbf{X})}{p_A^{NEE}(\mathbf{X})},$$

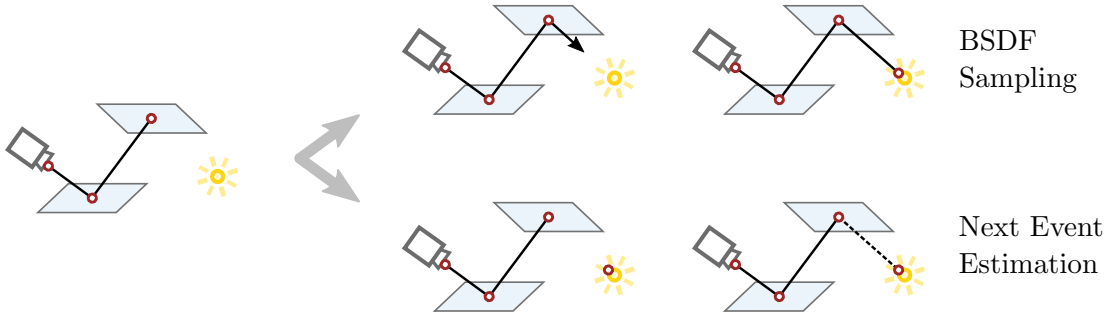


Figure 4.12: Both path tracing and NEE can turn a path prefix (**left**) into a full light transport path (**right**): Either we sample a direction from the BSDF that happens to lead to the light source (**top**), or we explicitly connect to a vertex on the light (**bottom**). Dotted lines indicate paths that were formed by connecting two vertices.

with MIS weights w_{NEE} and w_{PT}. If we use the balance heuristic (equation 4.3.5)

$$w_{NEE}(\mathbf{X}) = \frac{p_A^{NEE}(\mathbf{X})}{p_A^{NEE}(\mathbf{X}) + p_A^{PT}(\mathbf{X})} \qquad w_{PT}(\mathbf{X}) = \frac{p_A^{PT}(\mathbf{X})}{p_A^{NEE}(\mathbf{X}) + p_A^{PT}(\mathbf{X})},$$

the common factors of the two PDFs cancel out and we end up with

$$w_{NEE}((\mathbf{x}_0, \ldots, \mathbf{x}_k)) = \frac{p_A^{NEE}(\mathbf{x}_k)}{p_A^{NEE}(\mathbf{x}_k) + p_A^{BSDF}(\mathbf{x}_k)}$$
$$w_{PT}((\mathbf{x}_0, \ldots, \mathbf{x}_k)) = \frac{p_A^{BSDF}(\mathbf{x}_k)}{p_A^{NEE}(\mathbf{x}_k) + p_A^{BSDF}(\mathbf{x}_k)}.$$

This only works properly if both PDFs that we input to the balance (or power) heuristic have the same measure. In practice, they often start out in different measures – usually, BSDF sampling natively outputs a solid-angle PDF while NEE uses surface-area PDFs – but we can easily convert one of them to the other measure using equation 4.2.3.

4.5.3 Light Tracing

Light tracing basically works like path tracing with NEE, but in reverse (see figure 4.13). We start paths by sampling a light vertex and an emitted direction, and then build the path vertex by vertex, by iteratively tracing a ray to find the next vertex and then sampling a new direction from the BSDF at that vertex. Since we do not have a geometric camera model and instead assume an infinitely small camera, light tracing as we use it cannot hit a camera vertex by chance. Instead, we explicitly connect each scene vertex to the camera through NEE in order to form full light transport paths.

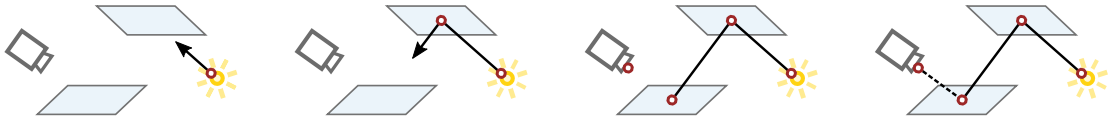


Figure 4.13: Light tracing (LT) constructs paths by sampling a point and direction on a light source (**left**), and then iteratively tracing rays (**center left**), before finally sampling (**center right**) and connecting to (**right**) a vertex on the camera.

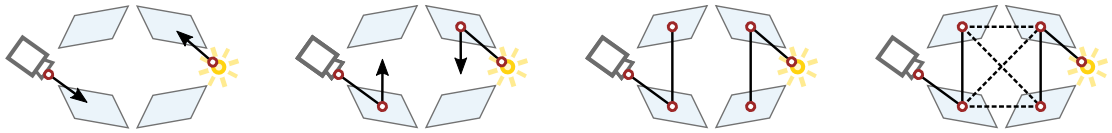


Figure 4.14: BDPT constructs paths by constructing a camera sub-path with path tracing and a light sub-path with light tracing, and then connecting the path vertices.

4.5.4 Bidirectional Path Tracing

Bidirectional path tracing (BDPT) combines path tracing and light tracing, and takes the idea of NEE one step further by connecting arbitrary vertices of the paths resulting from PT and LT. The basic idea is outlined by figure 4.14 and the following algorithm:

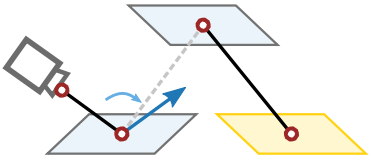
Algorithm 1: Bidirectional Path Tracing

```
F_N = 0
for n=1...N do
    PT trace path (x_0, ..., x_c)
    LT trace path (x_{c+1}, ..., x_k)
    for s=0...c do
        for t=c+1...k-1 do
            connect sub-paths to form x = (x_0, ..., x_s, x_t, ..., x_k)
            throughput = measurement(x) / PDF(x)
            w = MISweight(x, s, t)
            F_N += w · throughput
F_N /= N
```

We trace one *camera sub-path* with PT starting from the camera, and one *light sub-path* with LT starting from a light source. Then we iterate all possible connections between vertices from one path to the other, evaluate the measurement contribution and PDF of the resulting path, and compute the MIS weight before accumulating everything in the estimator.

To de-correlate the resulting paths we do not connect the camera path vertices to the same vertex on the light (nor light path vertices to the same camera vertex), and instead perform proper NEE for each of the vertices, such that each camera (or light) sub-path vertex gets connected to an independent NEE vertex on a light source (or the camera).

Connections to specular surfaces. Similar to how NEE does not work for connecting specular surfaces to light sources, BDPT does not work for connections to or from a specular surface, as illustrated on the right. If either end vertex of the camera or light sub-path is on a specular surface, the previous sub-path vertex uniquely determines the specular direction (blue) for which the surface scatters light, which can never align with the connection direction (dotted gray) to an independently sampled sub-path.



Multiple Importance Sampling

Thanks to the connections between camera and light sub-paths there now exist $k+1$ sampling strategies to sample a path $(\mathbf{x}_0, \ldots, \mathbf{x}_k)$ with $k+1$ vertices[8]. We distinguish these strategies by the number s and t of vertices that were sampled by PT and LT, respectively. For instance, the path $(\mathbf{x}_0, \ldots, \mathbf{x}_k)$ can be sampled by pure PT ($s = k+1, t = 0$), PT with NEE ($s = k, t = 1$), LT with NEE ($s = 1, t = k$) or any connection of a camera sub-path with $s > 1$ vertices to a light sub-path with $t > 1$ vertices. The PDFs for these strategies are as follows:

$$p_A^{s,t}((\mathbf{x}_0, \ldots, \mathbf{x}_k)) = \prod_{i=0}^{s} p_A^{cam}(\mathbf{x}_i) \cdot \prod_{i=t}^{k} p_A^{light}(\mathbf{x}_i),$$

where $p_A^{cam}(\mathbf{x}_i)$ and $p_A^{light}(\mathbf{x}_i)$ are the PDFs for sampling $\mathbf{x}_i$ during path tracing given the path prefix $(\mathbf{x}_0, \ldots, \mathbf{x}_{i-1})$, or during light tracing given the path suffix $(\mathbf{x}_{i+1}, \ldots, \mathbf{x}_k)$. The balance heuristic MIS weight for a path sampled with strategy s, t is

$$w_{s,t}((\mathbf{x}_0, \ldots, \mathbf{x}_k)) = \frac{p_A^{s,t}((\mathbf{x}_0, \ldots, \mathbf{x}_k))}{\sum_{i=0}^{k} p_A^{i,k+1-i}((\mathbf{x}_0, \ldots, \mathbf{x}_k))}.$$

4.5.5 Discussion

The four techniques presented in this section (PT, PT with NEE, LT with NEE, and BDPT) have different strengths and weaknesses with respect to the path types and image features they can sample well or even at all. We also discuss common alternatives, i.e. photon mapping and Markov Chain Monte Carlo methods, that address the remaining weaknesses in exchange for bias.

Path tracing has the advantage of being conceptually simple and easy to implement, and is able to sample all paths except those that hit a singular light: This would require the continuous (camera or BSDF) sampling leading up to the (point or directional) light to sample the singular direction that matches the singular light direction, which is theoretically impossible. The main disadvantage of PT is its high variance caused by the complete disregard for the incident radiance, as we only ever perform local BSDF sampling to extend the path and are thus unlikely to find a light source.

Path tracing with next event estimation solves this problem and generally results in much lower variance whenever the light source is hard to find by chance. However, NEE cannot sample paths with a specular surface before any kind of light source, since connecting a vertex on a specular surface to any continuously sampled vertex in the scene results in a connection direction that is guaranteed to be different from the singular true specular direction determined by the previous path vertex. Unfortunately this means that NEE cannot render caustic paths, where light gets focused or reflected to cast light patterns onto some other surface.

[8] If the camera model has actual geometry in the scene, i.e. if it is possible to hit the camera with a ray during light tracing, there are $k+2$ strategies. We do not support this, so we only count $k+1$.

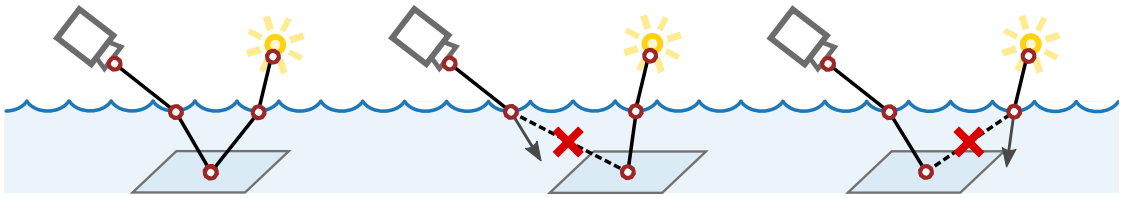


Figure 4.15: BDPT cannot sample SDS paths, such as paths looking through a specular water surface (**left**). With independently sampled camera and light sub-paths, it is theoretically impossible that the end vertex of the light sub-path aligns with the specular direction required by the last camera sub-path vertex (**center**), and vice versa (**right**).

Light tracing with next event estimation converges quickly for caustics (as opposed to path tracing, where caustics are noisy, and path tracing with NEE that cannot render caustics at all), but does not support connections from specular vertices to the camera. In practice we can combine light tracing, path tracing and next event estimation with MIS to handle a wider range of phenomena at once, but without the costly inner connections and complex implementation of bidirectional path tracing.

Bidirectional path tracing performs well wherever any of the previous methods work well, and offers additional benefits for some effects where the difficulty is not in the start or end of the path but somewhere in the middle. It still struggles with specular - diffuse - specular sequences as in figure 4.15: Similar to how NEE cannot connect a specular scene vertex to a light vertex, we cannot connect a camera and light sub-path if any of them ends in a specular vertex, as it is theoretically impossible for the connecting direction to align with the required specular direction. A major drawback of BDPT is its slow computation time for evaluating all the different connections and PDFs, especially if the majority of them is unnecessary for good performance in a given scene.

Photon mapping [33] and its derivatives [34] [35] [36] trace multiple light paths in a preprocessing pass and insert all light path vertices in a spatial data structure, the photon map, to store their location, incident direction and accumulated measurement contribution. During the actual rendering pass they start camera paths as with conventional path tracing, but instead of extending paths until they reach a light source they perform a density estimation by merging the last camera sub-path's vertex with the light path vertices from the photon map. This type of regularization is conceptually similar to the one we use in chapter 7. Photon mapping is able to quickly resolve of caustics, including those hidden inside SDS paths, but also introduces bias which manifests in multiple types of visual artifacts [37].

Metropolis light transport [38] [39] applies Markov Chain Monte Carlo integration and samples correlated paths by applying permutations to existing complete light transport paths. This enables quick exploration of difficult regions in path space such as SDS paths and other caustics. However, this exploration only occurs if we find a difficult region with an unbiased baseline renderer. Given finite time constraints we cannot guarantee this, which may result in missing image features and temporal instability.

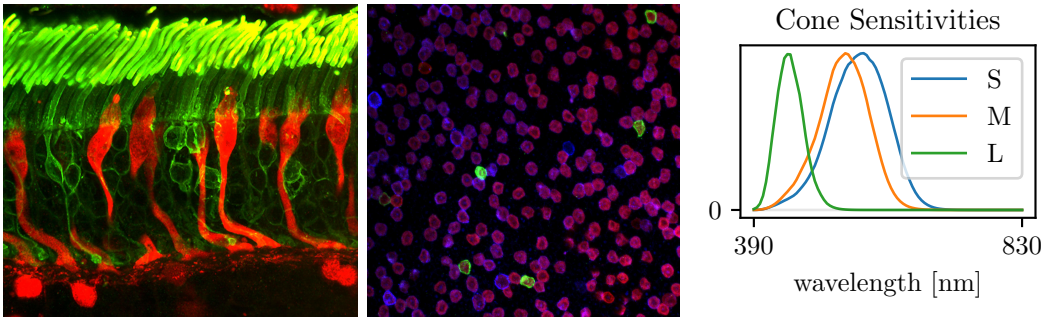


Figure 4.16: **Left**: "A confocal microscope image of rod and cone photoreceptors in a human retina. Fluorescent probes have been used to identify rod photoreceptors (green) and cone photoreceptors and horizontal cells (red). The cones looks like candle sticks, warped as if viewed through a fun house mirror. The horizontal cells are the round cells in the lower portion of the image." [48]. **Center**: "A picture of six cone cells that detect blue light (colored green) surrounded by a sea of cone cells that detect green light (colored red and blue)." [49]. **Right**: 10° cone spectral energy sensitivities [50].

4.6 Spectral Light Transport

A number of physically based rendering systems throughout academia and industry simulate the underlying light transport spectrally, i.e. by considering the wavelength dependency of radiometric quantities. This has a number of advantages: In contrast to simple RGB-based simulations, it becomes considerably easier to support wave-optical effects, such as iridescence due to thin films, scratches, or other types of rough surface microstructure [40, 41, 42]. In the context of visual effects, spectral rendering is used to replicate the spectral sensitivity curves of cameras, thereby recreating the same metamerism as the on-set objects to achieve a consistent look of real and virtual objects throughout all lighting conditions [43],[44]. Finally, spectra are the natural domain for computing products of colors (as is e.g. necessary to compute indirect illumination), and the approximation of this behavior through RGB triplets is often poor [45, 46].

How we perceive colors through a human eye depends on how the S-, M- and L-cones (a type of light-sensitive cell, figure 4.6) in our eyes react to the visible part of the spectrum, which is generally assumed to be somewhere [9] between 400nm and 700nm [24, ch. 4.2]. This allows us to represent any color impression as a triple of *tristimulus values* corresponding to the three cones' responses. We can convert tristimulus values between different color spaces (section 4.6.1), and even reproduce a good portion of all visible colors by combining a fairly saturated red, blue and green light source. This motivated conventional RGB-rendering, where we simulate "red", "green" and "blue" light separately. We shortly discuss why this is problematic in terms of faithful color reproduction (section 4.6.2), the changes we need to make to the rendering equation (section 4.6.3) to properly address spectral light transport, and how to generate input data for spectral rendering from existing RGB data (section 4.6.4).

[9] The exact numbers in literature vary, and when rendering we technically want to consider the wavelength range that is visible to the camera sensor instead of a human observer, although usually the two ranges should be similar [47].

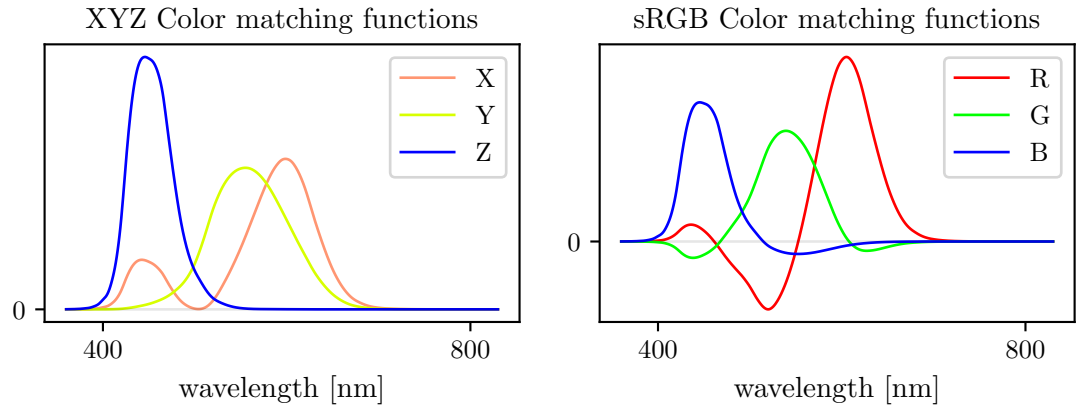


Figure 4.17: Color matching functions for the XYZ (**left**) and sRGB (**right**) color space. All three curves for the sRGB color space go below zero, meaning that some spectra cannot be represented with sRGB tristimulus values in $[0, 1]^3$.

4.6.1 Color Spaces

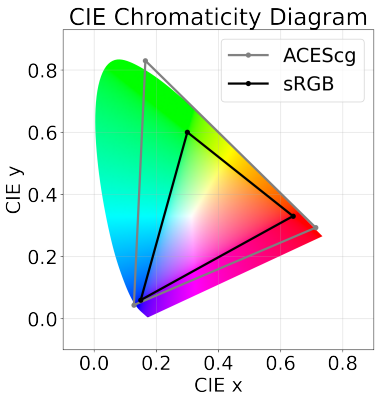
In this thesis we mainly use the RGB and XYZ color space, although we also introduce the xyY color space to better discuss chromaticity, and the CIELAB color space to better capture the perceived difference between two colors. Figure 4.17 shows the response curves for the XYZ and sRGB color space; xyZ and CIELAB values are usually constructed directly from XYZ triples. Since any color space basically reduces continuous spectra to three discrete responses (*tristimulus values*), different spectra can cause the same response and thus appear as the same color in an effect called *metamerism*.

The CIE 1931 XYZ color space (XYZ for short) is based on three imaginary primary colors that allow us to represent all visible colors with XYZ triples in $[0, 1]^3$. However, due to the imaginary primaries, more extreme XYZ values such as $(1, 0, 0)$ in turn produce imaginary colors. The Y-value corresponds to the color's luminosity, and all colors with linearly scaled XYZ values have the same chromaticity (apparent hue and saturation) at different luminosities.

The xyY color space separates a color's chromaticity (hue and saturation) from its luminance, by projecting an XYZ triple to the $X + Y + Z = 1$ plane:

$$x = \frac{X}{X + Y + Z} \qquad y = \frac{Y}{X + Y + Z}.$$

This allows us to plot color impressions in the *chromaticity diagram* on the right. In this diagram each point corresponds to a chromaticity xy, and the curved shape represents all visible chromaticities. The curious shape results from the response curves of the S-, M- and L-cones in our eyes.



RGB color spaces describe color impressions by combinig a red, green and blue primary color. The *gamut* of an RGB color space is the set of all color impressions that can be achieved by RGB triples in $[0,1]^3$, with more saturated primaries resulting in a bigger gamut. For instance, the gamut of the ACEScg color space [51] is much bigger than the gamut of the sRGB color space, as shown above. Due to the linear relationship between XYZ and all RGB color spaces, we can convert XYZ to RGB triples using a matrix, for instance to go from XYZ to sRGB we use [52]

$$M_{XYZ\to sRGB} = \begin{pmatrix} 3.2406 & -1.5372 & -0.4986 \\ -0.9689 & 1.8758 & 0.0415 \\ 0.0557 & -0.2040 & 1.0570 \end{pmatrix}.$$

Note how this matrix contains negative values; in fact this is the case for all RGB color spaces with non-imaginary primaries. This means that there are no three light sources that can be combined to reproduce all visible colors, and applying the respective conversion matrix to XYZ values for visible colors outside the respective RGB gamut results in negative RGB values.

The CIELAB color space [53] tries to tie the tristimulus values to perceptual properties, such that changes in the coefficients produce more uniform perceived color changes. The euclidean distance of two colors in this color space is a closer match to how we perceive the difference of two colors compared to their euclidean distance in an RGB color space. We use this to improve the fit of spectra to tristimulus values in section 5.5.

4.6.2 Problems with RGB Rendering

Conventional image synthesis relies on RGB rendering. With this technique, light transport paths are constructed independent of the wavelength of light, and their contributions are evaluated for a red, green and blue component. This discretization of a continuous spectrum into three separate values comes with several problems, two of which are indirect scattering colors and physically unreasonable reflectances. Hence the widely used practice of working with RGB values as units of computations in a renderer is actually wrong [46].

Incorrect scattered colors. In the real world, the spectrum $O(\lambda)$ of a beam of light that was scattered at some surface is the product of the spectrum $I(\lambda)$ of the incident beam and the reflectance spectrum $R(\lambda)$ at the surface[10]:

$$O(\lambda) = I(\lambda) \cdot R(\lambda).$$

RGB rendering computes the RGB values $O_{r,g,b}$ of the scattered beam of light directly from the RGB values $I_{r,g,b}$ for the incident beam and $R_{r,g,b}$ for the surface reflectance:

$$O_{r,g,b} = I_{r,g,b} \cdot R_{r,g,b}.$$

[10]Technically this reflectance spectrum depends on the incident and outgoing light directions; for simplicity we assume it is constant for any pair of directions. The reasoning in this section applies either way.

We can convert spectra $S(\lambda)$ to an RGB triple using the RGB space's color matching functions $C_{r,g,b}$ (see figure 4.17 for an example) via

$$R = \int_\Lambda S(\lambda)C_r(\lambda)\,\mathrm{d}\lambda \qquad G = \int_\Lambda S(\lambda)C_g(\lambda)\,\mathrm{d}\lambda \qquad B = \int_\Lambda S(\lambda)C_b(\lambda)\,\mathrm{d}\lambda.$$

However, converting the outgoing spectrum $O(\lambda)$ to an RGB triple generally yields a different result than directly multiplying the RGB values $I_{r,g,b}$ and $R_{r,g,b}$ of the incident and reflectance spectrum:

$$\int_\Lambda C_{r,g,b}(\lambda)I(\lambda)R(\lambda)\,\mathrm{d}\lambda \neq \int_\Lambda C_{r,g,b}(\lambda)I(\lambda)\,\mathrm{d}\lambda \cdot \int_\Lambda C_{r,g,b}(\lambda)R(\lambda)\,\mathrm{d}\lambda.$$

With additional scattering interactions between subsequent surfaces this effect only intensifies. The solution to this is spectral rendering (see section 4.6.3), where we sample and evaluate a wavelength with each light transport path, and only convert the spectrum to a tristimulus value once it arrives at the camera sensor.

Physically invalid reflectances. Real-world reflectance spectra of solid objects share some properties, such as following a fairly smooth shape and being restricted to no more than 100% reflectance of any given wavelength. Even with the second restriction alone the gamut of physically valid reflectances does not contain all visible colors, and not even some particularly bright and saturated colors in some of the larger RGB color spaces. The most saturated colors in this gamut lie on *MacAdam's Limit* and are produced by box-shaped spectra $\Lambda \to \{0, 1\}$. This gamut shrinks to the even smaller *gamut of natural reflectances* [54] if we require the spectra to follow a certain "smoothness" (see figure 5.2 in chapter 5 for example gamut and spectra). As a result, some of the RGB reflectances in RGB rendering may not have any matching spectra in the real world, taking this technique one step further away from physical realism. We address this issue in chapter 5.

Wavelength-dependent path geometry. In many scenes the path geometry depends on the wavelength: Dielectrics have a wavelength-dependent index of refraction and thus refract light rays of different wavelengths in different directions, iridescence produces vibrant colors due to interference of the light waves, and participating media have wavelength-dependent transmittances, meaning some wavelengths can travel much further through a medium before being absorbed. If we construct paths regardless of wavelength and only evaluate them for three channels, we cannot reproduce such effects.

4.6.3 Spectral Rendering

To describe how light travels from a light source to a camera sensor depending on its wavelength, we need to include the wavelength dependency in the description of light sources (section 4.4.2), BSDFs (section 4.4.3) and sensor sensitivity function (section 4.4.5), and extend the integral in the measurement equation (equation 4.4.4) to the wavelength domain.

We start from the wavelength-dependent radiance

$$L(\mathbf{x}, \omega, \lambda) = \frac{\mathrm{d}\Phi}{\mathrm{d}A(\mathbf{x})\,\mathrm{d}\sigma(\omega)\,\mathrm{d}\lambda}[\frac{W}{m^2 \cdot sr \cdot nm}]$$

and the wavelength-dependent irradiance

$$E(\mathbf{x}, \lambda) = \frac{\mathrm{d}\Phi}{\mathrm{d}A(\mathbf{x})\,\mathrm{d}\lambda},$$

and combine them in the wavelength-dependent BSDF

$$f_r(\omega_i, \lambda, \mathbf{x}, \omega_o) = \frac{\mathrm{d}L(\mathbf{x}, \omega_o, \lambda)}{\mathrm{d}E(\mathbf{x}, \lambda)} = \frac{\mathrm{d}L(\mathbf{x}, \omega_o, \lambda)}{L(\mathbf{x}, \omega_i, \lambda)\cos\theta_i\,\mathrm{d}\sigma(\omega_i)\,\mathrm{d}\lambda}.$$

Whenever we express the BSDF over vertices instead of directions we place the wavelength in the last argument, as in

$$f_r(\mathbf{z} \to \mathbf{y} \to \mathbf{x}, \lambda).$$

The spectral rendering equation. We describe the outgoing wavelength-dependent radiance at a surface as

$$L_o(\mathbf{x}, \omega_o, \lambda) = L_e(\mathbf{x}, \omega_o, \lambda) + \int_{\mathbb{S}^2} L_i(\mathbf{x}, \omega_i, \lambda) f_r(\omega_i, \lambda, \mathbf{x}, \omega_o) |\cos_{\mathbf{x}}\theta_i|\,\mathrm{d}\sigma(\omega_i),$$

and the pixel measurement equation

$$I^j = \int_{\mathcal{P}\times\Lambda} f^j(\mathbf{X}_\lambda)\,\mathrm{d}\mu(\mathbf{X}_\lambda).$$

describes the overall measurement at a pixel j as the integral over the wavelength-dependent path space $\mathcal{P} \times \Lambda$ containing all paths *and all wavelengths*. The measure in this integral is the product area and wavelength measure $\mathrm{d}\mu = \mathrm{d}A(\mathbf{x}_k)\ldots\mathrm{d}A(\mathbf{x}_0)\,\mathrm{d}\lambda$ consisting of the product area measure for the geometric path vertices $\mathbf{X} = (\mathbf{x}_0, \ldots, \mathbf{x}_k)$, and the Lebesgue measure for the path's wavelength λ.

The wavelength-dependent measurement contribution function $f^j(\mathbf{X}_\lambda)$ for pixel j is

$$\begin{aligned} f^j(\lambda, \mathbf{x}_0, \ldots, \mathbf{x}_k) =& W^j(\mathbf{x}_0, \mathbf{x}_1, \lambda) V(\mathbf{x}_0, \mathbf{x}_1) G(\mathbf{x}_0, \mathbf{x}_1) \\ & \cdot \prod_{j=1}^{k-1} f_r(\mathbf{x}_{j+1} \to \mathbf{x}_j \to \mathbf{x}_{j-1}, \lambda) V(\mathbf{x}_j, \mathbf{x}_{j+1}) G(\mathbf{x}_j, \mathbf{x}_{j+1}) \\ & \cdot L_e(\mathbf{x}_k \to \mathbf{x}_{k-1}, \lambda). \end{aligned}$$

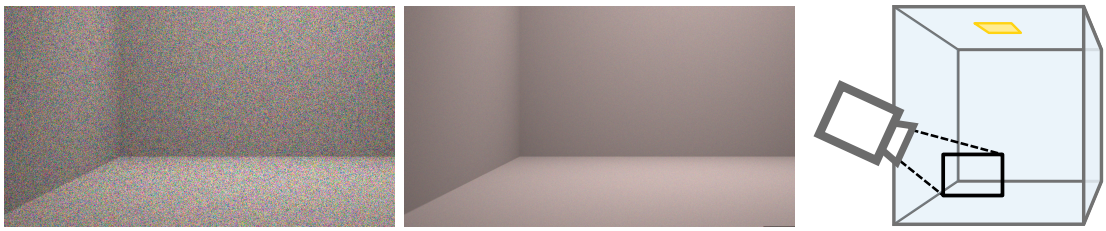


Figure 4.18: An empty gray box scene at 10 and 1000 samples per pixel. In principle this is easy to render, but exhibits considerable color noise at low sample counts (**left**) that vanishes in the long run (**right**).

Spectral path sampling

We can solve the wavelength-dependent measurement equation with Monte Carlo integration by sampling a wavelength and then sampling a path for that wavelength, for example with path tracing (section 4.5.1), light tracing (section 4.5.3) or bidirectional path tracing (section 4.5.4). With BDPT in particular we need to use the same wavelength for both the camera and light sub-path, such that the BSDFs at the connecting vertices have the same incident and exitant wavelength and are thus properly defined.

Sampling wavelengths on top of geometric paths introduces an additional dimension to our problem, and in turn causes additional variance that manifests as color noise, as shown by figure 4.18. This variance can be reduced by evaluating the contribution of a path for more than one wavelength [55] in a form of spectral multiple importance sampling. In simple configurations importance sampling in the spectral domain may also reduce variance, although in more complex scenes the spectrum arriving at the camera is not trivial to predict.

4.6.4 Spectral Upsampling

In order to perform spectral rendering we need spectra describing the emission at light sources, the reflectance at surfaces and the response of a camera sensor. However, most data for surfaces and lights in existing scene models is usually available in RGB, either as individual triples or in textures. For spectral rendering we need to somehow convert this RGB data to reflectance and emission spectra. This process is called spectral upsampling.

In chapter 5 we present a new upsampling scheme that includes fluorescent colors for particularly saturated RGB values that cannot be achieved through spectra limited at 100% reflectance, or that can only be achieved by reflectance spectra that are too "extreme" to be physically reasonable.

Previous work. There are many techniques available to upsample RGB values to a full spectrum. Other communities are interested in matching reference spectra (see for instance [56]). We briefly recapitulate the most relevant methods for graphics.

MacAdam introduced a family of box spectra that can model the brightest and most saturated reflectances [57]. Smits [58] uses optimization to generate spectra that are smooth and constrained to be smaller than one. Meng et al. [54] provide a full color gamut method for upsampling that also optimizes the smoothness of spectra. Energy conservation is not taken into account during the optimization and enforced later on

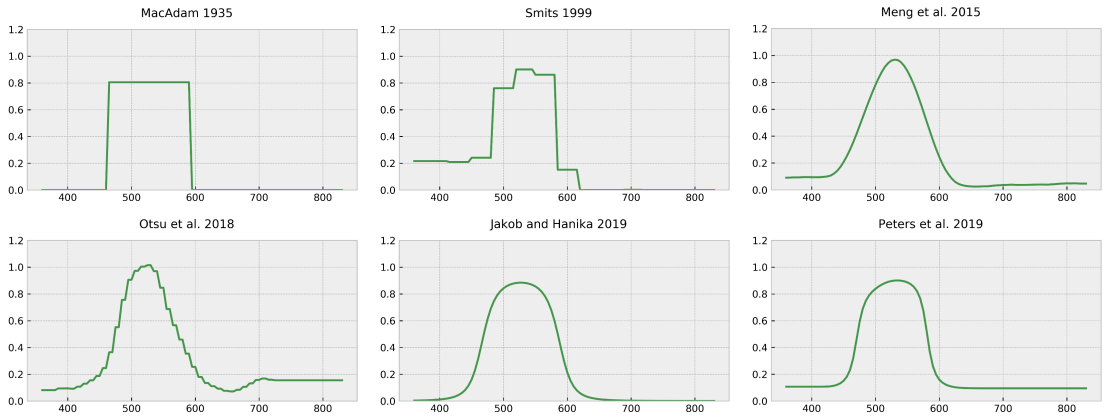
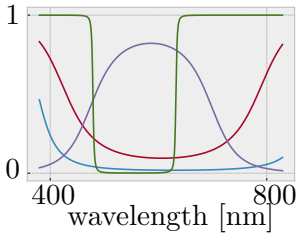


Figure 4.19: Upsampled spectra for the same RGB triple with different upsampling approaches [57] [58] [54] [59] [6] [60].

by scaling, hence bright spectra are not always achievable. Otsu et al. [59] use a clustered principal component analysis to reconstruct spectra from tristimulus input. Their method does not obey the $[0, 1]$ constraint of physically valid reflectance spectra. Peters et al. [60] employ a maximum entropy spectral estimate to reconstruct both reflectance and emission spectra from a variable number of Fourier coefficients. Figure 4.19 shows example spectra from various approaches.

The sigmoid spectra model. The parametric function space of Jakob and Hanika [6] is able to express both saturated colors using blocky spectra as well as more moderate input with smooth spectra (see figure on the right). The model is a composition of a quadratic polynomial and a sigmoid that ensures energy-conserving reflectance on the interval $[0, 1]$:



$$\begin{aligned} r(\lambda) &= S(P(\lambda)), \text{ with} \\ P(\lambda) &= c_0\lambda^2 + c_1\lambda + c_2 \\ S(x) &= \frac{1}{2} + \frac{x}{2\sqrt{1+x^2}}. \end{aligned} \tag{4.6.1}$$

The coefficients $c^r = (c_0, c_1, c_2)$ for an arbitrary RGB triple are interpolated from a 3D look-up table generated using a one-time offline optimization step. Due to the polynomial and sigmoid warp, a linear interpolation of the coefficients does usually not result in a linear color gradient. However, since both the colors and the coefficients of adjacent entries in the precomputed cube are fairly similar, this does not pose a problem in practice. When upsampling RGB textures, the three coefficients c^r are typically stored for each texel, replacing the prior tristimulus values.

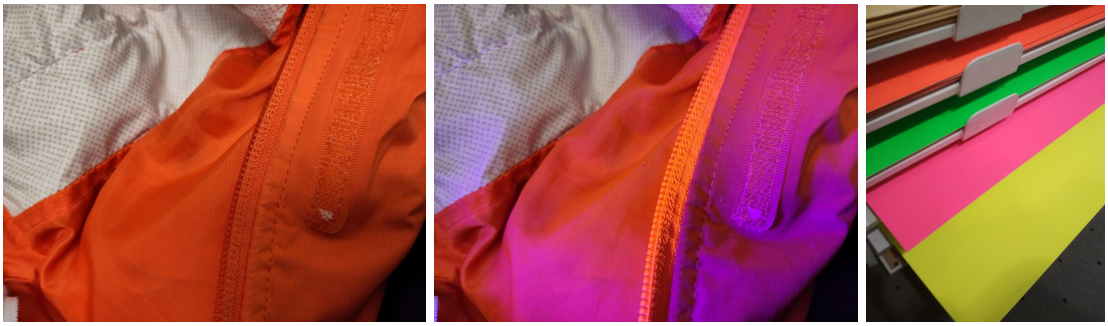


Figure 4.20: **Left**: Under normal indoor lighting, the zipper and fabric of a rain jacket appear in a similar color. **Center**: With an ultraviolet flashlight, fluorescence in the zipper makes it appear much brighter. **Right**: Fluorescent cardboard.

4.7 Fluorescence

Some substances can absorb a photon and after a brief moment re-emit a photon with another, typically longer wavelength. This can have different causes, one of which is fluorescence. Such processes are also called inelastic scattering [61], since the wavelength change is tied to a change in energy, opposed to elastic scattering at "ordinary" surfaces where light is scattered at the same wavelength with which it arrived.

In the real world, fluorescence occurs both in fluids (e.g. tonic water, ink in text markers, ...) and in solid surfaces, both natural (leaves, algae, teeth, ...) and artificial (fire trucks, paper, textiles (figure 4.20), ...). In product design, fluorescence causes a more intense and bright white for clothes or paper, more saturated colors for toys, or even better visibility in safety-related items such as safety vests or cones, so we should take fluorescence into account if we want to accurately reproduce the appearance of these materials. In biology [62] and chemistry [63], much work focuses on basic measurements of fluorescence in molecules, but also advanced applications such as fluorescent imaging, where fluorescent dyes are inserted into living cells. Most of this work is centered around solvents, where fluorescent molecules are dissolved in some fluid, and will help us understand the principles of fluorescence [63, 64].

We discuss the physical processes behind fluorescence (sections 4.7.1 and 4.7.2), how to model the collective scattering behavior of a fluorescent surface in the *bispectral bidirectional reflection and reradiation distribution function* (BBRRDF, section 4.7.3), how to extend the rendering equation to include this phenomenon (section 4.7.4) and what we need to consider when rendering scenes with fluorescence (section 4.7.5).

4.7.1 Background

The term "fluorescence" refers to a process where a photon is absorbed at one wavelength and re-emitted at another. The absorption causes the molecule to enter a so-called *excited state*, in which one of its outer electrons transitions to a higher energy level. This state is unstable, and after roughly 10^{-8} seconds the electron relaxes down to a lower energy level in multiple transitions, one of which is accompanied by emission of a photon corresponding to the energy difference. This is different from *phosphorescence*,

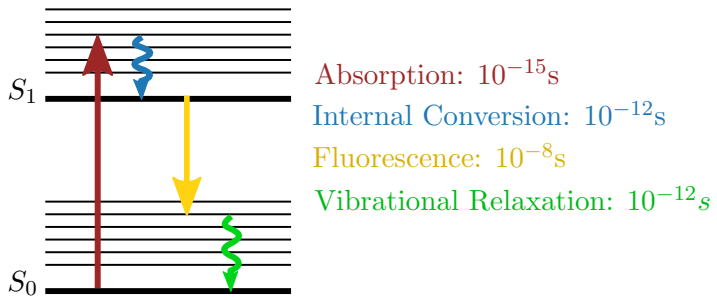


Figure 4.21: The Jablonski energy diagram shows the lowest levels of the singlet states as bold lines, and higher vibrational and rotational energy levels as thin lines (note that there are many more vibrational levels than depicted in the diagram [66]). Energy transitions involving absorption or emission of photons are depicted as straight arrows, while wavy arrows signify transitions that do not involve photons. After absorbing a photon elevates the molecule to an excited energy state (red, 10^{-15}s), internal conversion (blue, after 10^{-12}s) relaxes the molecule to the lowest energy level of the excited state S_1. After some 10^{-8}s, the molecule drops back to some random level in the ground state S_0 (yellow) while emitting a photon corresponding to the energy difference. This is what we call fluorescence. The final vibrational relaxation (10^{-14}s to 10^{-12}s [66]) quickly brings the molecule back to the lowest level of S_0 (green).

a process involving additional transitions to more stable energy states that usually takes in the order of seconds [63] [65]. In the context of rendering, we can consider fluorescence instantaneous, at least with typical camera shutter times of multiple milliseconds. Proper phosphorescence rendering usually requires considering time as an additional dimension.

The Jablonski energy diagram in figure 4.21 illustrates the energy levels of a molecule and the transitions between them. A molecule capable of fluorescence has multiple possible energy states: The ground singlet state S_0, one or more singlet states $S_1, S_2, ...$ with higher energy, and possibly a triplet state T_0 where we end up if the transitioning electron changes spin. At room temperature, the electron usually occupies one of the vibrational and rotational levels of the ground singlet state, and most likely is in the lowest level of the ground state [67, ch. 3.1].

If a photon with the right amount of energy hits the molecule, its energy is absorbed and the electron transitions to a higher vibrational and rotational level of one of the higher singlet states. The "distance" of this transition corresponds to the photon's energy. This state is highly unstable, and the electron relaxes to the lowest level of S_1 after roughly $10^{-12}s$ in a process called internal conversion [66]. The energy difference for this transition is lost as heat. After roughly $10^{-8}s$, the electron relaxes from S_1 to one of the vibrational and rotational levels of S_0. The energy difference depends on which level the electron relaxes to, and is emitted as light. Since there are many such levels, the emitted wavelength is not deterministic and instead distributed according to the transition probabilities of the individual levels in S_0. All light-emitting relaxations occur from the base level of S_1, so the emitted wavelength is independent of the absorbed wavelength.

In fluids fluorescence is typically diffuse, as a molecule can rotate multiple times while in the excited state [63, ch. 1.5], meaning that the direction in which the photon leaves the molecule is independent of the original incident direction as well. In more viscous solutions, or if the fluorescent molecule is bound in another way, the substance may exhibit some anisotropy in the polarization of the emitted light. For simplicity and in line with the observations by Wilkie et al. [68] we still assume fluorescence is diffuse, and note that extending to non-diffuse fluorescence simply requires replacing the directional component of the fluorescent reflectance model.

With phosphorescence the electron changes spin while dropping from the lowest level of S_1 in a process called intersystem crossing, where the molecule transitions to a triplet state without emitting a photon. As this state is more stable, the final light-emitting relaxation to S_0 only occurs after a much longer time frame in the order of seconds [63].

Measuring fluorescence. Not every photon whose energy could potentially excite a molecule is actually absorbed. Instead, the probability for fluorescent absorption depends on the wavelength and is represented by an *absorption spectrum*. On top of that, not all photons that were absorbed get re-emitted. Sometimes the energy is lost in other ways, e.g. as heat. This ratio of emitted photons to absorbed photons is called the *quantum yield*. The *emission spectrum* represents the distribution of emitted wavelengths. The two spectra are usually independent of each other [63, ch. 1.3.2]. The spectra and quantum yield can vary with physical circumstances such as temperature, the solvent, or even over time, if ongoing illumination changes the substance on a chemical level (*photobleaching*).

We can measure the shape of the absorption spectrum by measuring the emission at a constant wavelength while illuminating a substance with different wavelengths. Similarly, we can measure the emission spectrum by illuminating a substance with a single wavelength and measuring the emitted light over all wavelengths. In practice, the measured absorption and emission spectra often overlap (see figure 4.22). Assuming that the emitted wavelength is truly independent of the absorbed wavelength, this raises the question whether absorbed photons with wavelengths in this overlapping region can be re-emitted at shorter wavelengths, or if this is impossible since shorter wavelengths carry higher energy. We discuss this in section 4.7.2.

Typical properties. Most fluorescent substances share some typical properties. The brightest fluorescent dyes usually have spectra in the 300nm to 700nm range, as photons with wavelengths below 200nm have enough energy to ionize the molecule, while photons with longer wavelengths do not have enough energy to elevate most fluorophores to even the lowest excited states, and molecules with a low enough excited state to allow such absorption usually have other options of dissipating the absorbed energy [64].

Figure 4.22 shows a selection of fluorescent absorption and emission spectra from the fluorophores database of TU Graz [69]. All spectra follow a mostly smooth shape, with one to three peaks in the absorption spectrum and usually one peak in the emission spectrum. Most of the measured spectra range from 300nm to 800nm, matching the soft lower and upper wavelength bounds from Sauer et al. [64, ch. 1.1].

The *Stokes' shift* is the distance between the absorption spectrum's main peak and the emission spectrum's peak. It is bigger for some substances and smaller for others, but the emission spectrum's peak is always at a longer wavelength, and the Stokes'

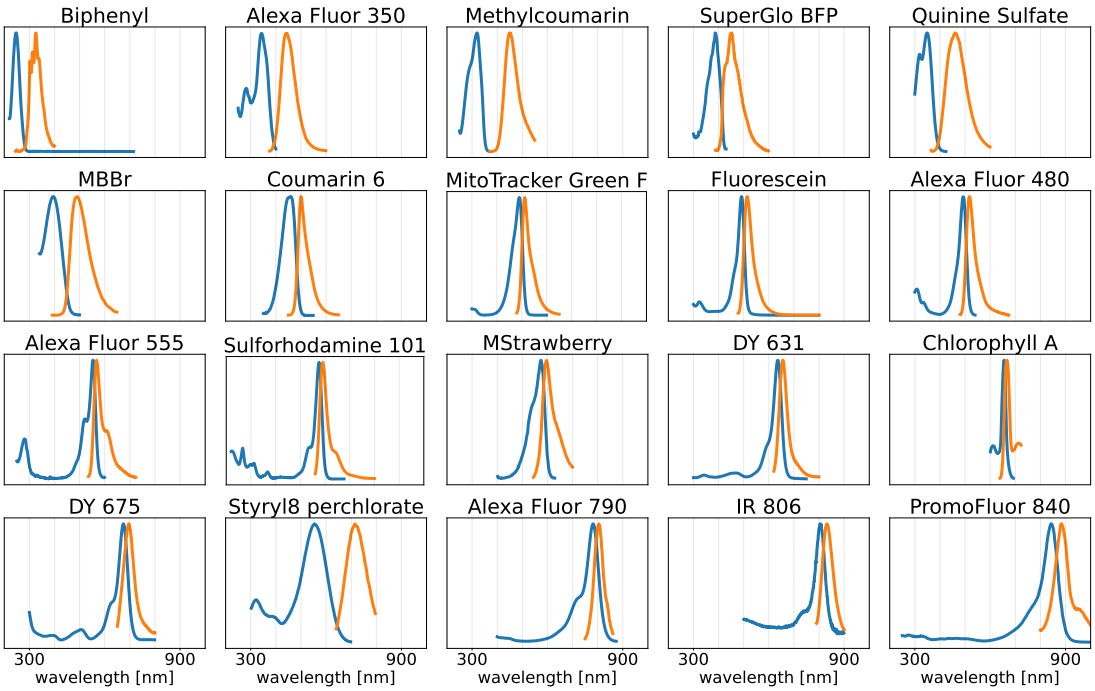


Figure 4.22: Fluorescent absorption (blue) and emission (orange) spectra measurements from the Fluorophores.org database [69].

shift ranges from a few to over a hundred nanometers. In most cases the emission spectrum roughly mirrors the final peak of the absorption spectrum, albeit with a slight distortion. This is called the mirror image rule and caused by the similar spacing between the vibrational energy levels in states S_0 and S_1 [66, 63].

The spectra in figure 4.22 were measured for fluid solutions and only quantify fluorescent aspects of the substance. Real-world surfaces with fluorescence, as the cardboard that we measured for figure 4.23, not only have a diffuse fluorescent component, but also contain some non-fluorescent component that preserves the wavelength of incident light during scattering [68]. The *reradiation matrix* [70] captures both fluorescent and non-fluorescent behavior in a single data structure, where entry $m_{i,j}$ of the matrix describes

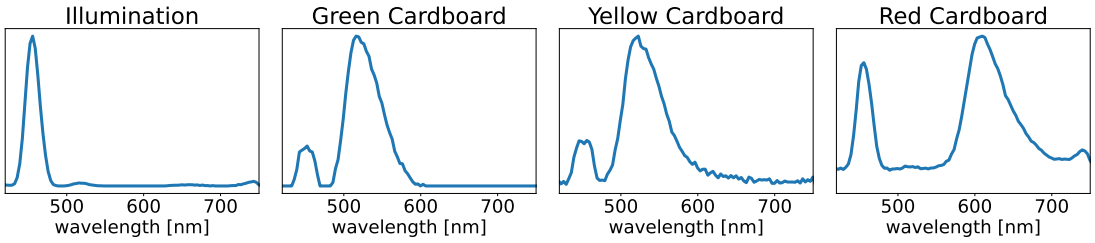


Figure 4.23: Measured outgoing radiance for a green (**left**), yellow (**center**) and red (**right**) cardboard from figure 4.20, illuminated with a filtered spectrum. The peak in the wavelength range of the incident light corresponds to non-fluorescent scattering, and the peak at longer wavelengths is caused by fluorescent absorption and re-emission.

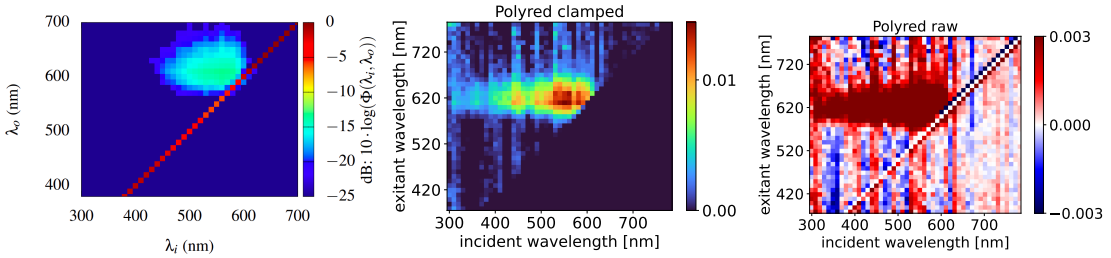


Figure 4.24: **Left**: Reradiation matrix from Mojzík et al. [73], who attribute the shorter-wavelength reradiation (below the diagonal) to measurement noise. **Center**: Reradiation matrix, as used by [74], who set the lower-diagonal entries to 0 and clamp negative values to 0. The diagonal is set to 0 as well so as to not overshadow the remaining entries. **Right**: Raw measurement data (`POLYRED.BFC` material from the RIT fluorescent database [75, 76]), clamped to $[-0.003, 0.003]$, diagonal set to 0. It is unclear whether the ringing next to the diagonal and the measurement noise hide a small region of shorter-wavelength reradiation.

how much energy arriving at wavelength λ_i gets scattered to wavelength λ_j, with the diagonal also including non-fluorescent scattering.

Many reradiation matrices in the literature (figures 4.24, 4.25, 4.27, 4.28) exhibit roughly linearly dependent rows and columns. This confirms that the emission spectrum's shape is mostly independent of the absorbed wavelength. We can also observe that most of the fluorescent re-emission happens on the side of the diagonal corresponding to shorter-to-longer-wavelength reradiation. Some measurements did not provide data for the other side of the diagonal, but in the remaining examples we can observe a small but non-zero reradiation from longer to slightly shorter wavelengths. Whether these values are caused by noise or measuring inaccuracies, or by actual reradiation at shorter wavelengths (section 4.7.2), is unclear.

4.7.2 Reradiation at Shorter Wavelengths

In this section we discuss whether fluorescent emission at shorter wavelengths than the absorbed wavelength is generally possible. On the one hand, this seems to contradict conservation of energy, as the energy of a photon is inversely proportional to its wavelength, and longer-to-shorter wavelength reradiation adds energy to a photon. On the other hand, absorption and emission spectrum measurements often overlap, and if the emission spectrum is truly independent of the absorbed wavelength (due to the relaxation to the lowest level of S_1 after any absorption event), what happens if a photon with a wavelength in this overlapping area is absorbed?

On top of that, some of the reradiation matrices in the literature, where longer-to-shorter-wavelength reradiation was not discarded in advance, exhibit some amount of reradiation on both sides of the diagonal (see figure 4.25), although it is not immediately clear if this can also be explained by noise or other measurement inaccuracies (e.g. off-diagonal ringing in figure 4.25 (left), crosstalk between neighboring filter bands [71, ch. 4.2]), or other sources of uncertainty [72, ch. 4.2]).

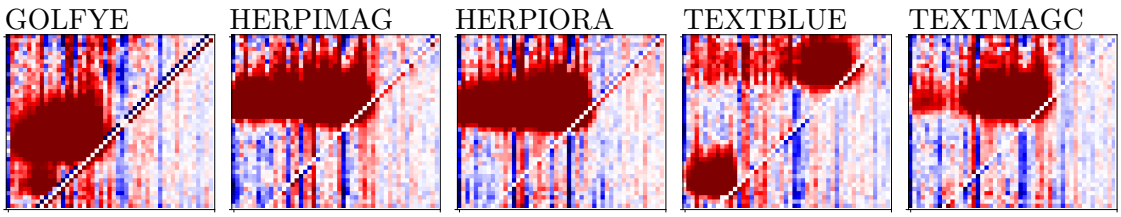


Figure 4.25: Reradiation matrices clamped to [-0.005,0.005] for incident wavelengths $[300, 780]$ nm and exitant wavelengths $[380, 780]$ nm, without the diagonal, for several materials from the RIT fluorescent data set [75, 76]. The red region extending below the diagonal suggests fluorescence from longer to shorter wavelengths.

Early experiments

Interestingly, this controversy reaches back as far as the mid 19th century, where researchers started investigating how fluorescence affects the spectrum of incident light. In 1991 Malley [77] compiled a detailed summary of this era of early experimental fluorescence research, in the following we only highlight some examples. In initial experiments Becquerel [78] observed that some materials scatter visible light when being illuminated with ultraviolet light[11]. Stokes [79] further observed that reradiation exclusively happens at longer wavelengths, and concludes

> *There is one law relating to internal dispersion which appears to be universal, namely, that when the refrangibility of light is changed by dispersion it is always lowered.* [79, page 490]

This was before the term "fluorescence" took hold; here, "internal dispersion" simply means fluorescence, and "refrangibility" relates to how strongly the light's wavelength is refracted, so a changing refrangibility means a changing wavelength.

From today's point of view we take his conclusion with a grain of salt, since Stokes' method of observation was to move fluorescent samples through a rainbow generated by a series of prisms, and look at them through sheets of colored glass. Him not observing any shorter wavelength reradiation might also have other explanations – as the overlap between absorption and emission spectra is small to nonexistent for many materials, any potential shorter-wavelength reradiation might have been too weak or narrow-banded to be noticeable, mixed up with slightly longer wavelength reradiation, or even reradiated to longer wavelengths after a second fluorescent interaction.

[11] Quote: *Un fait que j'ai souvent observé dans le cours de ces expériences, c'est que, lorsque le spectre solaire frappe une substance telle que la sulfure de calcium phosphorescent, pendant son action le papier sur lequel est déposée cette substance parait lumineux non-seulement de A en H, mais encore jusqu'en P; de sorte qu'on peut voir toutes les raies du spectre de A en P dessinées par projection.* [78, p. 320] Loose translation: Some substances such as calcium sulfide appear luminous not only for illuminating rays from spectral categories A (extreme red) to H (extreme violet), but also up to category P (ultraviolet illumination).

Nonetheless, this hypothesis became known as *Stokes' Law*, followed by years of researchers investigating its universality [80, 81] [12]. Due to experimental inaccuracies and difficulties in producing sufficiently strong, narrow-banded illuminations[13], several results observing reradiation at shorter wavelengths were soon dismissed as inconclusive. For instance, in Stenger's [80] discussion of previous experiments, he notes that depending on the method, some observations of shorter wavelength reradiation may have also been caused by other sources, but judges the experimental setup by Lommel [82, p. 251-252] with a prism followed by two glass plates to filter sunlight as sufficiently conclusive. He then verifies shorter wavelength reradiation for multiple substances with his own setup using prisms, lenses and movable slits to isolate narrow spectral bands.

More recent reradiation matrix measurements mostly ignore potential shorter wavelength reradiation, some since their measurement setup is not precise enough [83], and some arguing that such reradiation is not possible and should be set to zero [74] or not measured at all [71].

Modern Explanations

With the rise of quantum theory and more insights in the vibrational energy levels of molecules came an explanation for the energy gain with shorter-wavelength reradiation:

> *such an "energy defect" is compensated for (as stated by Einstein for the first time) by the fact that at room temperature, a small fraction of molecules are in a vibrational level higher than level 0 (distribution among the energy levels fulfilling the Boltzmann law [...]) in the ground state as well as in the excited state. At low temperature, this departure from the Stokes law should disappear.* [84, chapter 3.1.2, page 57]

The authors support this with multiple observations of a noticeable reradiation at shorter wavelengths [85].

Other work uses the term "anti-Stokes" for fluorescent emission at shorter wavelengths [86, 87]. Anti-Stokes fluorescence can have different causes, such as the molecule starting in an already excited state from heat (as described by the Boltzmann distribution of the molecule), corresponding to the long tail of the absorption spectrum, or simultaneous excitation by two long-wavelength photons [87] [63, ch. 1.11.1] [88]. This also implies that with increasing heat, the distribution of initial energy levels has increasingly more molecules starting at higher levels. At room temperature most (but not all!) molecules start in the lowest possible energy state [67, ch. 3.1][66], and increasing the temperature results in more molecules starting from a higher excited state [77] as well as in a broader absorption spectrum [64, ch. 1.6].

[12] Quote: *Damit ist unzertrennlich die Frage nach der Allgemeingültigkeit des Stokes'schen Gesetzes verbunden, für dessen unbedingte Richtigkeit Hagenbach eintritt, während nach Lommel, Lubarsch, Brauner demselben nur eine beschränkte Anwendbarkeit ankommt.* [80] Translation: This is inextricably linked to the question of the universal validity of Stokes' law, which Hagenbach advocates as being unconditionally correct, while according to Lommel [82], Lubarsch, and Brauner it only has limited applicability.

[13] For instance, certain experiments had to be delayed until the next summer, as they required sufficiently intense sun light [81].

The anti-Stokes shift plays a big part in modern applications, such as fluorescent cooling [89], where solids, gases and liquids cool down because of energy loss from fluorescent shorter-wavelength reradiation, or allowing fluorescent imaging to use infrared light which is able to penetrate deeper into a sample before interacting with a fluorescent dye [86, 87].

What does this mean for rendering?

In the context of rendering, we expect that supporting or ignoring reradiation at shorter wavelengths leads to only negligible differences: First, the overlap between a typical absorption and emission spectrum covers only a fraction of the emission spectrum's energy. Removing this part of the emission spectrum hardly changes the color of the emitted spectrum to begin with. Second, in most rendering settings fluorescent surfaces are illuminated by a continuous spectrum, so usually most of the incident radiance has wavelengths shorter than the overlapping range. Plus, since the overlap only covers a fraction of the absorption spectrum distribution, shorter wavelengths are absorbed much more likely than wavelengths in the overlapping range. Overall, the reradiated spectrum will generally have a very similar color in either case.

In the end, we decided to treat such reradiation as possible, both because of the existing support in early experimental literature [80] [84] [85] and more recent anti-Stokes applications [89, 86], but also since the color difference in practice is usually close to nonexistent. This simplifies modeling of fluorescent BBRRDFs (see section 4.7.3), and any rendering technique that works for overlapping absorption and emission also works for the subset of cases without reradiation at shorter wavelengths.

4.7.3 Fluorescent BBRRDF

We describe fluorescent surfaces with the *bispectral bidirectional reflection and reradiation distribution function* (BBRRDF) [71]:

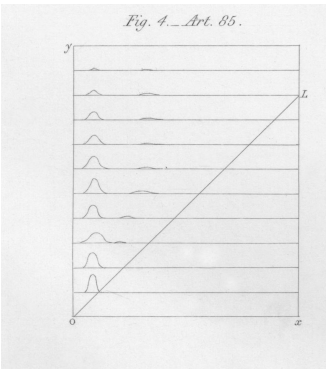
$$f(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o) = \frac{\mathrm{d}^2 L_o(\mathbf{x}, \omega_o, \lambda_o)}{L_i(\mathbf{x}, \omega_i, \lambda_i) \cos\theta_i \, \mathrm{d}\sigma(\omega_i) \, \mathrm{d}\lambda_i}.$$

BBRRDF models can be fully data-driven by measuring the BBRRDF from a material sample, partially data-driven by combining an analytical BRDF with measured fluorescence data (either in the form of a reradiation matrix [70] [71], or in the form of an absorption and emission spectrum [8]), or fully analytical.

Energy conservation. A BBRRDF is energy conserving if it fulfills the condition

$$\forall\omega_i \forall\lambda_i \int_\Lambda \int_\Omega f(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o) \, \mathrm{d}\omega_o^\perp \, \mathrm{d}\lambda_o \leq 1.$$

If the BBRRDF is based on a reradiation matrix with rows corresponding to exitant wavelengths and columns corresponding to incident wavelengths (as in figure 6.6), energy conservation means that no column may sum up to more than 1. Not all energy conserving BBRRDFs are photon conserving, especially for reradiation toward longer wavelengths, as a photon's energy is inversely related to its wavelength.



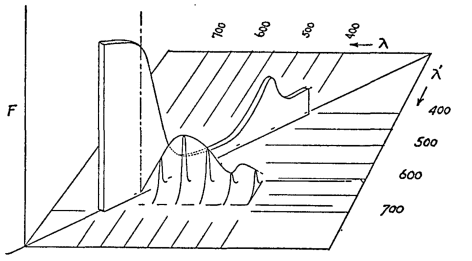


FIG. 10. Perspective representation of the type of fluorescing material discussed. λ is the wavelength of the incident beam, λ′ is the wavelength of the light sent back. The reflectance curve, drawn on the diagonal, is shown having finite thickness for the sake of clarity, actually in this representation it should be a portion of a plane.

Figure 4.26: **Left**: Matrix-like layout depicting the qualitative behavior of a solution of leaf-green by Stokes in 1852 [79, figure 4], Quote: "Fig. 4 is a rude representation [...] merely intended to assist the reader in forming a clear conception of the general nature of the phenomena, and must not be trusted for details". **Right**: Matrix-like plot of early fluorescence spectra measurements by Tyler et al. in 1951 [90, figure 10].

Reflexion of fluorescent ink (red)

Incident light (mμ)	*Spectral reflexion (%) mμ*											
	300	340	380	420	460	500	540	580	620	660	700	740
300	3·5							1 9	5 6	2·1	0·5	0·1
340		6·7						1 6	5·8	2·2	0·6	0·1
380			6 8					2 3	7·0	2·5	0·8	0 2
420				2·8				2 9	8·6	3·2	1 0	0·2
460					4·4			2 7	8·3	3 4	0 8	0 2
500						4·3		3·0	8·3	3 4	0 8	0·3
540							2·4	3 7	8·8	3 6	0 9	0 3
580								8 0	9·3	3 9	1 0	0 4
620									79 5	0·4	0 1	—
660										84·1	—	—
700											85·1	—
740												86·0

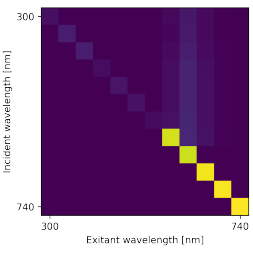


Figure 4.27: Reradiation matrix measured by Donaldson in 1954, original data (**left**, [83, p. 212]) and visualization (**right**).

Acquisition. Previous work in natural sciences resulted in numerous data sets of measured absorption and emission spectra [76, 69, 91, 92]. Unfortunately these data sets are of limited use for rendering surfaces, since the absorption and emission properties depend on the specific solvent used in the measurement, which generally differ from those for the material composition of a solid surface. In addition, real-life fluorescent surfaces also have a non-fluorescent component [72, p. 171] that is missing from these measurements, as well as directional variations as the reflected light not only depends on incident and exitant wavelength but also on incident and exitant direction.

Detailed measurements from surfaces that are applicable to rendering are rare, since they are time consuming to perform and memory intensive to store. Ideally, we would want a full reradiation matrix with measurements for all combinations of incident and exitant wavelengths and directions, meaning a 6D measurement ($2 \times$ 2D directions and $2 \times$ 1D wavelengths) for a spatially uniform surface and 8D for spatially varying surfaces. Early works use a matrix-like layout for a more qualitative discussion of fluorescence, as in figure 4.26. Donaldson [83] was one of the first to measure and report a full reradiation matrix (see figure 4.27), although the data has a fairly low spectral resolution at 12 40nm-wide steps from 300nm to 740nm, and ignores anti-Stokes fluorescence (in other words, the matrix do not contain data below the diagonal) and directional variation.

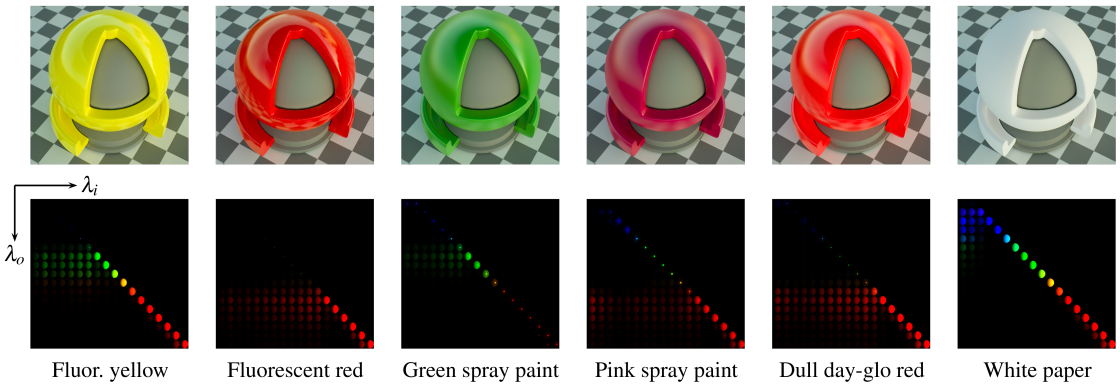


Figure 4.28: Figure 1 from the BBRRDF measurements performed by Hullin et al. [71]. The matrix entries show miniature rendered spheres at the given incident/exitant wavelength pair. Note that the axes are swapped compared to figure 4.27.

Most notably, Hullin et al. [71] performed full BBRRDF measurements for a number of materials (see figure 4.28). Their measurements include both spectral and directional dependencies with a higher spectral resolution of 20nm steps, but are not publicly available.

Instead of performing costly measurements that fully resolve the bispectral domain, recent research tries to reconstruct fluorescent data from a few sparse measurements, for example reconstructing spectra from RGB images using principal component analysis [93] or joint optimization of fluorescence and reflectance [94], only considering fluorescence induced by UV light which is then measured with a single UV light source [95], or reconstructing reflectance and emission spectra from hyperspectral images with different illuminants [96][97]. Such reconstructions do yield data – either in form of a reradiation matrix or as a pair of absorption and emission spectra – that can be used in a fluorescent renderer, but because of the sparse input data and underlying heuristics they are not necessarily in line with the ground truth.

Modeling. Wilkie et al. [68] proposed the first analytical models for fluorescent surfaces. They observe that typical fluorescent surfaces have both a non-fluorescent and a fluorescent component, with the fluorescent component behaving mostly diffuse, while the non-fluorescent component can be more glossy to specular. For their model, they use existing BRDF models that can be separated into a diffuse and glossy to specular component, and apply a reradiation matrix to the diffuse part.

Storing a full reradiation matrix requires quadratic memory compared to storing a (non-fluorescent) reflectance spectrum. In previous work[14] [8] we rely on the observation that the emission spectrum is typically independent of the absorption spectrum and that at low enough concentrations fluorescence is roughly linear [98], and parameterize the diffuse fluorescent component with the two spectra instead of a full matrix, plus two scalar values $c, Q \in [0, 1]$ corresponding to the fluorescent concentration and quantum

[14]This is not part of the contributions in this thesis.

yield:

$$f_f(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o) = \frac{c \cdot a(\lambda_i) \cdot Q \cdot e(\lambda_o)}{\pi}.$$

In principle this can be combined with any arbitrary spectral BSDF f_r via[15]

$$f_{fr}(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o) = f_f(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o) + \delta_{\lambda_o}(\lambda_i)(1 - ca(\lambda_i))f_r(\omega_i, \mathbf{x}, \omega_o, \lambda_i), \quad (4.7.1)$$

although throughout this thesis we only use a diffuse non-fluorescent BRDF

$$f_r(\omega_i, \mathbf{x}, \omega_o, \lambda) = r(\lambda)/\pi$$

parameterized by a reflectance spectrum $r(\lambda)$.

The spectra and reradiation matrix in these models can be measured or analytical. We derive a phenomenological fluorescence spectrum model in chapter 5 that only needs to store three scalar coefficients instead of the two fluorescence spectra, and use it to reproduce appearance under a given illuminant. Hua et al. [74] fit a Gaussian mixture model to measured reradiation matrices to reduce memory overhead, requiring more storage than the analytical spectra in exchange for being able to accurately reproduce appearance of measured materials under any illumination.

4.7.4 Extended Rendering Equation

In order to accurately simulate fluorescence we rely on spectral rendering, i.e. simulating and evaluating light-surface-interactions for a given wavelength instead of constructing a path independent of its wavelength and evaluating it for a number of discrete color channels, as in traditional RGB rendering or discretized spectral rendering [70] [99] [100]. The bispectral rendering integral [71]

$$L_o(\mathbf{x}, \omega_o, \lambda_o) = \int_\Omega \int_\Lambda L_i(\mathbf{x}, \omega_i, \lambda_i) f_f(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o) \cos\theta_i \, \mathrm{d}\sigma(\omega_i) \, \mathrm{d}\lambda \quad (4.7.2)$$

describes the exitant radiance at some surface $\mathbf{x}$ due to fluorescent absorption and reemission described by the BBRRDF f_f. Note that this only includes fluorescent scattering. We include non-fluorescent scattering with a non-fluorescent component f_r as well as emitted light in the extended rendering equation

$$\begin{aligned} L_o(\mathbf{x}, \omega_o, \lambda_o) = & L_e(\mathbf{x}, \omega_o, \lambda_o) \\ & + \int_\Omega \int_\Lambda L_i(\mathbf{x}, \omega_i, \lambda_i) f_f(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o) \cos\theta_i \, \mathrm{d}\lambda \, \mathrm{d}\sigma(\omega_i) \\ & + \int_\Omega L_i(\mathbf{x}, \omega_o, \lambda_o) f_r(\omega_i, \mathbf{x}, \omega_o, \lambda_o) \cos\theta_i \, \mathrm{d}\sigma(\omega_i). \end{aligned}$$

As this equation contains two integrals, recursively inserting L_o for L_i as in equation 4.4.6 results in a rather clumpy expression. We use the shorthand notation

$$\int_\Lambda \delta_{\lambda_o}(\lambda) r(\lambda) \, \mathrm{d}\lambda := r(\lambda_o)$$

[15] Note that the δ-Notation is a purely symbolic abbreviation. It means that if f_{fr} is used in an integral over Λ, the integral needs to be replaced by a sum, i.e. $\int_\Lambda f_f(\lambda) + \delta_{\lambda_o}(\lambda)(1 - ca(\lambda))f_r(\lambda) \, \mathrm{d}\lambda := \int_\Lambda f_f(\lambda) \, \mathrm{d}\lambda + (1 - ca(\lambda_o))f_r(\lambda_o)$. Section 4.7.4 shows how to use this in the rendering integral.

as a purely symbolic abbreviation, which allows us to write the non-fluorescent part as

$$\int_\Omega L_i(\mathbf{x}, \omega_o, \lambda_o) f_r(\omega_i, \mathbf{x}, \omega_o, \lambda_o) \cos\theta_i \, \mathrm{d}\sigma(\omega_i) =$$
$$\int_\Omega \int_\Lambda L_i(\mathbf{x}, \omega_i, \lambda_i) \delta_{\lambda_o}(\lambda_i) f_r(\omega_i, \mathbf{x}, \omega_o, \lambda_o) \cos\theta_i \, \mathrm{d}\lambda \, \mathrm{d}\sigma(\omega_i),$$

and express the combined reflectance as

$$f_{fr}(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o) = f_f(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o) + \delta_{\lambda_o}(\lambda_i) f_r(\omega_i, \mathbf{x}, \omega_o, \lambda_o)$$

for a shorter notation of the extended rendering equation

$$L_o(\mathbf{x}, \omega_o, \lambda_o) = L_e(\mathbf{x}, \omega_o, \lambda_o) + \int_\Omega \int_\Lambda L_i(\mathbf{x}, \omega_i, \lambda_i) f_{fr}(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o) \cos\theta_i \, \mathrm{d}\lambda \, \mathrm{d}\sigma(\omega_i).$$

Again, note that the δ-notation is purely symbolic and indicates that we actually deal with two integrals: one over $\Omega \times \Lambda$ for f_f, and one over Ω for f_r.

Recursively applying this equation as in equation 4.4.6 gives us the measurement equation for a pixel j:

$$\begin{aligned} I^j = \sum_{k=1}^{\infty} \int_{\mathcal{M} \times (\Lambda \times \mathcal{M})^k} & W(\mathbf{x}_0 \leftarrow \mathbf{x}_1, \lambda_1) V(\mathbf{x}_0, \mathbf{x}_1) G(\mathbf{x}_0, \mathbf{x}_1) \\ & \cdot \prod_{j=1}^{k-1} f_{fr}(\mathbf{x}_{j+1}, \lambda_{j+1}, \mathbf{x}_j, \lambda_j, \mathbf{x}_{j-1}) V(\mathbf{x}_j, \mathbf{x}_{j+1}) G(\mathbf{x}_j, \mathbf{x}_{j+1}) \\ & \cdot L_e(\mathbf{x}_k \to \mathbf{x}_{k-1}, \lambda_k) \, \mathrm{d}A(\mathbf{x}_k) \, \mathrm{d}\lambda_k \ldots \mathrm{d}\lambda_1 \, \mathrm{d}A(\mathbf{x}_0), \end{aligned}$$

where technically the k-th integral over $\mathcal{M} \times (\Lambda \times \mathcal{M})^k$ is an abbreviation for the sum of 2^{k-1} integrals corresponding to all possible sequences of fluorescent and non-fluorescent interactions at vertices $\mathbf{x}_1, \ldots, \mathbf{x}_{k-1}$.

4.7.5 Rendering Fluorescence

Path tracing. Rendering fluorescent surfaces with path tracing and NEE is relatively straightforward, with the main difference that we write paths as $(\mathbf{x}_0, \lambda_1, \mathbf{x}_1, \ldots, \lambda_k, \mathbf{x}_k)$ with one wavelength λ_j per segment $(\mathbf{x}_{j-1}, \mathbf{x}j)$. For convenience we still write all segment wavelengths even if some of the vertices do not fluoresce, i.e. if $\lambda_i = \lambda_{i+1}$ we still write $(\ldots, \lambda_i, \mathbf{x}_i, \lambda_{i+1}, \mathbf{x}_{i+1}, \ldots)$ instead of $(\ldots, \lambda_i, \mathbf{x}_i, \mathbf{x}_{i+1}, \ldots)$, even though the path is technically part of the space $\mathcal{M} \times (\Lambda \times \mathcal{M})^i \times \mathcal{M} \times (\Lambda \times \mathcal{M})^{k-i-1}$, but including this case distinction at every vertex results in a rather blown up notation.

Similar to ordinary spectral rendering, we first sample a wavelength λ_0 at the camera with PDF $p_{camera}(\lambda_1)$. Then, whenever we hit a non-fluorescent surface, we keep tracing the path as we usually would, and keep the current wavelength. Whenever we hit a fluorescent surface, we sample a new wavelength with which to continue the path. The variance of this step can be reduced by importance sampling the new incident wavelength from the absorption spectrum, or the column / row (depending on how we orient the axes) in the reradiation matrix corresponding to the outgoing wavelength.

Hybrid surfaces are treated just like non-fluorescent surfaces with specular and non-specular components. First comes a discrete sampling decision deciding whether to continue with a non-fluorescent or fluorescent interaction (similar to deciding between a specular and non-specular interaction). Second, in case of a fluorescent interaction, we sample a new wavelength, and in case of a non-fluorescent interaction we keep the current wavelength.

We write the PDF of such a path as

$$p_A(\mathbf{x}_0, \lambda_1, \ldots, \lambda_k, \mathbf{x}_k) = \prod_{i=0}^{k} p_A(\mathbf{x}_i) \cdot \prod_{i=1}^{k} p_\Lambda(\lambda_i)$$

and set

$$p_\Lambda(\lambda_i) = \begin{cases} p_{camera}(\lambda_0) & \text{if } i = 1 \\ P_{fluo}(\lambda_i|\lambda_{i-1}) \cdot p_{absorption}(\lambda_i|\lambda_{i-1}) & \text{if } x_i \text{ fluorescent and } \lambda_i \neq \lambda_{i-1} \\ P_{nonfluo}(\lambda_i|\lambda_{i-1}) & \text{if } x_i \text{ fluorescent and } \lambda_i = \lambda_{i-1} \\ 1 & \text{if } x_i \text{ not fluorescent,} \end{cases}$$

where P_{fluo} and $P_{nonfluo}$ are the probabilities for selecting a fluorescent or non-fluorescent interaction.

This means that the measure of a PDF for a path with k segments now depends on the number of fluorescent interactions (in other words, the number of wavelength changes) along a path. Another interpretation of this is that whenever we construct a path and hit a potentially fluorescent surface $\mathbf{x}_i$, where the scattered radiance consists of both fluorescent and non-fluorescent scattering

$$L_o(\omega_o, \lambda_o) = \int_\Omega \int_\Lambda L_i(\omega_i, \lambda_i) f_f(\omega_i, \lambda_i, \lambda_o, \omega_o) \, \mathrm{d}\lambda + L_i(\omega_i, \lambda_o) f_r(\omega_i, \omega_o, \lambda_o) \, \mathrm{d}\sigma^\perp(\omega_i),$$

we first importance sample which of the two summands $\int_\Lambda L_i f_f \, \mathrm{d}\lambda$ and $L_i f_r$ to evaluate, followed by sampling a new wavelength in case of a fluorescent interaction. The first decision is a discrete Monte Carlo estimate of a finite sum and does not affect the measure of the path's PDF.

Overall, this situation is equivalent to handling perfectly specular mirrors or transmission, where the incident direction determines the scattered direction deterministically, collapsing the PDF of the next direction to a singularity and reducing the measure of the combined path PDF. In practice, we unify fluorescent and non-fluorescent sampling just as we would unify non-specular with specular sampling, by replacing the corresponding wavelength (or directional) PDF with 1 for purely non-fluorescent (or specular) surfaces, or with the selection probability $P_{nonfluo}$ of choosing the non-fluorescent (or specular) component over the fluorescent (or non-specular) one.

Path tracing with fluorescent volumes. In this thesis we focus on fluorescent surfaces, since path tracing with fluorescent volumes poses additional challenges. Since path tracing essentially back-traces light transport paths from the camera to the light, whenever we trace a path for a for wavelength that falls within a medium's fluorescent emission spectrum, we need to sample collisions in the medium where the wavelength could have been emitted due to fluorescence. Conventional volume distance sampling is based on the transmittance, which will likely not match the fluorescent emission behavior and result in poor importance sampling, and will even fail to sample these collisions completely if the medium is transparent to wavelengths in the fluorescent emission spectrum. Thus accurately rendering fluorescent volumes requires either the use of a light tracer, or specialized fluorescence-aware volumetric distance sampling schemes [73]. Abdellah et al. [101] also simulate fluorescence in media, and the fluorescence of natural waters has been considered [102].

Previous approaches. The first discussion of fluorescence rendering by Glassner [70] sets up the fluorescent rendering equation and highlights the similarities to phosphorescence. He uses a reradiation matrix to describe surfaces, but only allows Stokes fluorescence (meaning shorter-to-longer wavelength reradiation). Since hardware limitations at the time prohibited the use of Monte Carlo spectral sampling as it is done today, he replaces the three channels in an existing RGB renderer with a more refined discretization of the spectral domain, and uses the restriction to Stokes fluorescence to justify iterative rendering of one pass per wavelength band, sorted from short to long wavelengths, turning surfaces with fluorescent emission into area light sources in later passes. This was followed by research into combining fluorescence with polarization [103] and photon mapping [104], and tracing of individual photons [105] in the context of predicting the efficiency of fluorescence-based solar concentrators.

For typical camera shutter times fluorescence can be assumed to be instantaneous. Nalbach et al. [106] investigate phosphorescence, which is similar to fluorescence except that it can occur over up to several seconds, which requires additional care.

In more recent work, Mojzík et al. [73] extend a path tracer to enable importance sampling of fluorescent volumes and combine fluorescence with hero wavelength sampling to reduce spectral noise. Jarabo et al. [100] introduce fluorescence to a bidirectional context by tracing vectors of wavelengths instead of only one wavelength per segment, although their solution does not support wavelength-dependent directional sampling. In chapter 6 we use regularization to enable bidirectional path tracing to simultaneously handle fluorescence and arbitrary directional scattering.

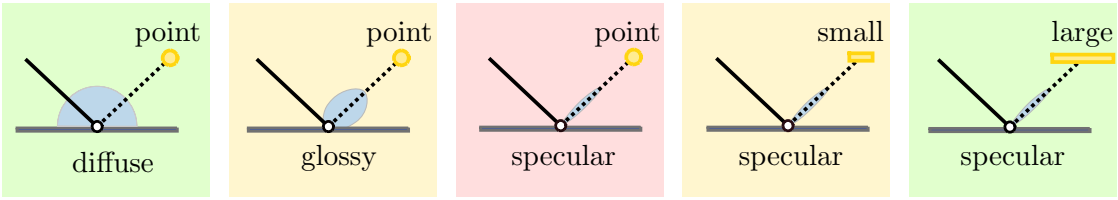


Figure 4.29: **Green**: Configurations with one singularity and one smooth, well-behaved integrand are easily sampleable by either NEE (**left**) or BSDF sampling (**right**), since forcing the direction to follow the singularity often produces a significant contribution from the smooth component. **Yellow**: Configurations with one singularity and one near-singularity are hard to sample and cause more noise, since the singularity forces a direction where the near-singularity may only have a small contribution. **Red**: Two subsequent singularities are impossible to sample with both NEE and BSDF sampling.

4.8 Singularities in the Rendering Equation

In section 4.4.6 we introduced the rendering integral

$$L_s(\mathbf{x}, \omega_o) = \int_{\mathbb{S}^2} L_i(\mathbf{x}, \omega_i) f_r(\omega_i, \mathbf{x}, \omega_o) \cos\theta_i \, \mathrm{d}\sigma(\omega_i)$$

to describe how light scatters at surfaces, by integrating the product of the incident radiance L_i with the surface's BSDF f_r. All of these symbols represent functions – L_i and L_s represent the incident and scattered radiance as a function over surfaces and directions, and f_r represents the ratio of exitant radiance over differential incident irradiance as a function over incident directions, surface vertices, and outgoing directions.

However, certain types of surfaces (e.g. mirrors or glass) and light sources (e.g. point lights or directional lights) cannot be described with a BSDF or radiance function. In fact, these effects cannot be described by the rendering integral at all, even if we were to allow arbitrary functions for f_r or L_i that are not proper BSDFs or radiance functions.

For instance, consider a mirror, where the outgoing radiance in direction ω_o is the incident radiance from the reflected direction ω_{or}, scaled by the mirror's albedo $a(\omega_o)$:

$$L_s(\mathbf{x}, \omega_o) = a(\omega_o) L_i(\mathbf{x}, \omega_{or}).$$

There is no BSDF to describe such surfaces since the BSDF is defined as the derivative of outgoing radiance with respect to irradiance

$$f_r(\omega_i, \mathbf{x}, \omega_o) = \frac{\mathrm{d}L(\mathbf{x}, \omega_o)}{\mathrm{d}E(\mathbf{x}, \omega_i)} = \frac{\mathrm{d}L(\mathbf{x}, \omega_o)}{L(\mathbf{x}, \omega_i) \cos\theta_i \, \mathrm{d}\sigma(\omega_i)},$$

but in case of a mirror, the outgoing radiance is not differentiable at the reflected direction. Even if we allow functions that are not a BSDF for f_r we cannot solve this issue, as there exists no function that we could insert for f_r in the rendering integral to achieve this behavior.

We call such configurations *singularities*, as the parts of the integrand collapse, and we only evaluate the non-singular remainder of the integrand at a single discrete point, instead of integrating the full integrand over a continuous domain.

Singular integrands are impossible to sample with continuous sampling strategies, resulting in missing features in the rendered image if no other sampling strategy is available. For instance, path tracing cannot sample paths ending in a point light, since sampling a direction from a continuous BSDF will never hit a single discrete point (figure 4.29 left). Similarly, next event estimation cannot sample paths with a mirror before the light source, since connecting a vertex on a mirror to a vertex on a light will never match the reflected direction of the outgoing direction at the mirror vertex (figure 4.29 right).

Other types of surfaces (e.g. rough mirrors or small light sources) result in an integrand with most or all of its mass inside a rather small support. We call such integrands "peaky" or "near-singular". These are sampleable in theory, but in practice only a small fraction of samples actually lands within a range that evaluates to a significant contribution, if the PDF does not match the integrand's peak. This causes strong noise in the image, leading to long convergence times. For instance, small area lights are often missed by directions sampled from a BSDF, and next event estimation with glossy, near-specular surfaces often results in connecting directions outside the BSDF's glossy peak (figure 4.29 yellow).

The definition of a "peaky" integrand is intentionally vague. This raises the question when a peaky integrand turns "smooth" enough to be sampled with reasonably low noise. In practice, this is determined by a user-defined threshold [107], or solved implicitly by combining sampling strategies with multiple importance sampling (MIS) [17].

We discuss different mathematical models for singularities in sections 4.8.1 and 4.8.2, and show how to use them to describe common singular scene features. In section 4.8.3 we introduce mollification, a form of regularization, where we replace a singular or near-singular integrand with a *mollifier* – a family of more well behaved functions that enable easier integration and converge to the true solution in the limit. This introduces bias, but turns singularities into sampleable problems, and reduces noise for peaky integrands that could technically already be sampled, albeit only with high variance.

4.8.1 Singularity Models

Many light transport effects in the real world can be described by integrating an incident radiance field multiplied to some scattering operator (e.g. a BSDF or phase function). Some effects, however, do not depend on a continuous incident radiance field or scattering function, but only on incident "light"[16] from a single direction, or only scatter light for a discrete set of parameters (directions, positions, wavelengths, ...). In other words, some effects require expressions in the form of

$$I = f(a) \cdot b$$

in a framework that usually works with expressions such as

$$I = \int f(x)g(x)\,\mathrm{d}\mu(x).$$

[16]Only considering a discrete direction or position affects the physical units. This means that technically we may no longer have (incident) radiance and need to deal with incident "light" in some other form.

In terms of integration, we can view singularities as a collapsed integral, where the support of the integrand collapses to a single point in the integration domain. Since we usually integrate with continuous measures (i.e. $\sigma^\perp, \sigma$ and A), this support is now a zero set, and integrating something that is only supported by a zero set should result in an integral with value 0. As this is obviously not helpful for singularities, we need to look for different mathematical models so we can properly describe such singular behavior. The remainder of this section proposes several such models, and shows how to apply them to common singular scene features.

Model 1: Singularities without an Integral

As singular effects cannot be described by a traditional integral, we can either modify the integral in some way, or describe them without using an integral at all. For instance, for a point light in direction ω_p at distance d, the directly scattered radiance at a scene vertex $\mathbf{x}$ only depends on the BSDF at $\mathbf{x}$ and the point light's radiant intensity I_p $[W/sr]$ (equation 4.4.2):

$$L_s(\mathbf{x}, \omega_o) = \frac{I_p}{d^2} f_s(\omega_p, \mathbf{x}, \omega_o) \cos\theta_p. \tag{4.8.1}$$

Note that we compute outgoing *radiance* by multiplying incident *radiant intensity* in $[W/sr]$, a BSDF in $[1/sr]$, and a cosine, and dividing by a squared distance in $[1/m^2]$. The radiant intensity is Watt over steradian, but with steradian centered at the point light. The multiplication with $\cos\theta_p/d^2$ transforms this to Watt over surface area incident at $\mathbf{x}$, resulting in the overall irradiance at $\mathbf{x}$. Multiplying this with a BSDF in $[1/sr]$ results in $[W/(m^2 sr)]$, which means that the scattered light can now be described as radiance, even though we start with incident light in the form of radiant intensity.

For mirrors (or specular surfaces in general) with albedo $a(\omega_o)$ the scattered radiance only depends on radiance from the reflected (or specular) direction ω_{or}:

$$L_s(\mathbf{x}, \omega_o) = L_i(\mathbf{x}, \omega_{or}) a(\omega_o). \tag{4.8.2}$$

In this case, we have proper radiance on both sides of the equation, since the incident radiance evaluated for the reflection is only scaled by a unitless albedo.

The scattered radiance at fluorescent surfaces with BBRRDF f_{fluo} can be expressed as an integral over both directions and wavelengths:

$$L_s(\mathbf{x}, \omega_o, \lambda_o) = \int_{\mathbb{S}^2} \int_{\Lambda} L_i(\mathbf{x}, \omega_i, \lambda_i) f_{fluo}(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o) \cos\theta_o \, \mathrm{d}\lambda_i \, \mathrm{d}\sigma(\omega_i).$$

In this context, non-fluorescent BSDFs pose a singularity, as they only scatter incident radiance toward a single wavelength. We can represent this behavior without the integral over wavelengths, which simply degenerates to the classical spectral rendering equation (equation 4.4.5).

These integral-free representations are straightforward to construct and read, and do not rely on any additional symbolic notations. The singular behavior is made explicit, not hidden behind some generalized BSDF, measures, or additional symbols. However, properly capturing all light transport in a scene now requires explicit case distinctions for every vertex and every light source where we sum up the singular and non-singular components. This results in a cluttered notation for longer path segments or multiple singular components.

Model 2: Singularities as a Dirac integrand

The Dirac notation is a common approach to unite both singular and non-singular behavior in a single symbol:

$$\int_D \Phi(x)\delta_a^\mu(x)\,\mathrm{d}\mu(x) := \begin{cases}\Phi(a) & \text{if } a \in D \\ 0 & \text{else.}\end{cases} \tag{4.8.3}$$

This notation pretends that there exists a special "function" δ_a^μ that we can multiply to an integrand Φ such that the overall integral evaluates to the non-singular integrand Φ at one fixed argument a. This allows us to unite singular and non-singular behavior in a single symbol that can be replaced by either a continuous function or a Dirac δ_a^μ, depending on the current context. However, as this "function" is tied to the integral's measure μ, the integral over δ_a^μ is only defined if the integral's measure is the same as the Dirac's measure. This requires special treatment when changing the measure of the integrand, and whenever we have more than one singularity at once.

Dirac integrand for specular surfaces. Starting from the rendering equation

$$L_s(\mathbf{x}, \omega_o) = \int_{\mathbb{S}^2} L_i(\mathbf{x}, \omega_i) f_r(\omega_i, \mathbf{x}, \omega_o) \cos\theta_i \,\mathrm{d}\sigma(\omega_i)$$

we can treat the symbol f_r as a sort of generalized BSDF acting as a placeholder for either an ordinary BSDF for non-singular surfaces, or a Dirac $a(\omega_o)\delta^\sigma_{\omega_{or}}(\omega_i)$ for singular surfaces that scatter from direction ω_{or} to ω_o with albedo $a(\omega_o)$. This allows for a convenient notation unifying both singular and non-singular surfaces, since inserting the Dirac results in

$$\int_{\mathbb{S}^2} L_i(\mathbf{x}, \omega_i) a(\omega_o)\delta^\sigma_{\omega_{or}}(\omega_i) \cos\theta_i \,\mathrm{d}\sigma(\omega_i) = L_i(\mathbf{x}, \omega_{or}) a(\omega_o) \cos\theta_{or},$$

which is exactly what we want for specular surfaces (also see equation 4.8.2).

Dirac integrand for point lights. Similarly, starting from the rendering equation over surface area measure

$$L_s(\mathbf{y} \to \mathbf{z}) = \int_{\mathcal{M}} L_o(\mathbf{x} \to \mathbf{y}) f_r(\mathbf{x} \to \mathbf{y} \to \mathbf{z}) G(\mathbf{x}, \mathbf{y}) \,\mathrm{d}A(\mathbf{x})$$

we can treat $L_o(\mathbf{x} \to \mathbf{y})$ as a sort of generalized exitant light representation, which we replace with the outgoing radiance at $\mathbf{x}$ for ordinary area lights, and with $I_p\delta^A_{\mathbf{p}}(\mathbf{x})$ for point lights at position $\mathbf{p}$ with intensity I_p. Assuming[17] the cosine with the point light inside $G(\mathbf{p}, \mathbf{y})$ is 1, this resolves to

$$\int_{\mathcal{M}} I_p\delta^A_{\mathbf{p}}(\mathbf{x}) f_r(\mathbf{x} \to \mathbf{y} \to \mathbf{z}) G(\mathbf{x}, \mathbf{y}) \,\mathrm{d}A(\mathbf{x}) = I_p f_r(\mathbf{p} \to \mathbf{y} \to \mathbf{z}) G(\mathbf{p}, \mathbf{y}),$$

which is exactly what we want for point lights (also see equation 4.8.1), as the geometry term includes the squared distance falloff.

[17] This begs the question what the normal of a point is supposed to be. One answer to this is that we can express the behavior of a point light as the limit of a family of shrinking disk or sphere lights that do have a normal which aligns with $\mathbf{p} \to \mathbf{y}$, in which case the cosine between the normal $\mathbf{p} \to \mathbf{y}$ and direction $\mathbf{p} \to \mathbf{y}$ does evaluate to 1.

Dirac integrand for non-fluorescent surfaces. In the context of fluorescent surfaces, where we compute the outgoing radiance as

$$L_s(\mathbf{x}, \omega_o, \lambda_o) = \int_{\mathbb{S}^2} \int_{\Lambda} L_i(\mathbf{x}, \omega_i, \lambda_i) f_{fluo}(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o) \cos\theta_i \, \mathrm{d}\lambda_i \, \mathrm{d}\sigma(\omega_i),$$

non-fluorescent surfaces pose a singularity. In this case we can replace f_{fluo} with $f_r(\omega_i, \mathbf{x}, \omega_o, \lambda_o)\delta^{\lambda}_{\lambda_o}(\lambda_i)$, i.e. the non-fluorescent BSDF f_r multiplied to a Dirac with Lebesgue measure on the wavelength domain Λ. This resolves to the (non-fluorescent) spectral rendering integral:

$$\begin{aligned} &\int_{\mathbb{S}^2} \int_{\Lambda} L_i(\mathbf{x}, \omega_i, \lambda_i) f_r(\omega_i, \mathbf{x}, \omega_o, \lambda_o)\delta^{\lambda}_{\lambda_o}(\lambda_i) \cos\theta_i \, \mathrm{d}\lambda_i \, \mathrm{d}\sigma(\omega_i) \\ =&\int_{\mathbb{S}^2} L_i(\mathbf{x}, \omega_i, \lambda_o) f_r(\omega_i, \mathbf{x}, \omega_o, \lambda_o) \cos\theta_i \, \mathrm{d}\omega_i. \end{aligned}$$

Issues with the Dirac integrand. Having both a specular surface and singular light within either form of the rendering integral (solid angle measure or surface area measure) is problematic. In this case, only one of the Diracs disappears when resolving the integration, and the other Dirac remains in the solution for the scattered light L_s. This is not defined until we use it in another integral, and even then the other integral's measure has to match the measure of the remaining Dirac. We discuss this in more detail in appendix C.1.2.

On top of that, the two examples just assumed that we need a Dirac with respect to solid angle measure σ for singular BSDFs, and a Dirac with respect to surface area measure for singular light sources. But how do we know that the Diracs should be tied to these particular measures in the first place? We either need to know the result we want in advance, so we can choose the measure and Dirac such that the integral resolves to that particular result. Or we can model singularities as the limit of increasingly peaky functions, in which case the integration domain and measure arise naturally. We discuss this in the context of distributions in section 4.8.2.

Model 3: Singular Measures

The value of an integral depends on three things: The integration domain, the integrand, and the measure. Using the Dirac symbol as described above changes the integrand for singular phenomena. Instead, we can move the scattering behavior and/or incident light to the measure, and then insert either a singular or a continuous measure depending on the actual scene configuration. In appendix A.1 we show how to base the measure for singular components on the Dirac measure (equation 4.1.1), and construct measures corresponding to the BSDF or incident radiance function for non-singular components.

The advantage of this model is that singular measures are actually perfectly well defined in the integration and measure framework from section 4.3, we do not need to worry about tying a Dirac integrand to the correct measure, and we do not end up with an undefined value for L_o in case of multiple singularities. We can even solve such integrals with Monte Carlo integration, by constructing a PDF that integrates to 1 with respect to our new extended measure. In the non-singular case this PDF works as usual,

while in the singular case with Dirac measure δ_x we can use PDFs like

$$p(y) = \begin{cases} 1 & \text{if } x == y \\ 0 & \text{else.} \end{cases}$$

While such functions integrate to 0 with a continuous measure, with the matching Dirac measure they now integrate to 1 and can thus be used as a valid PDF for importance sampling integrals with a Dirac measure. This justifies the common approach to "sample" new directions at specular surfaces by computing a deterministically reflected or refracted direction and setting its PDF in the Monte Carlo estimator to 1.

Summary

All three approaches – describing singularities without an integral, with a symbolic Dirac integrand, or with a properly defined Dirac measure – come with some sort of case distinction. With the integral-free description this distinction is explicit since we have different equations for singular and non-singular configurations. The symbolic Dirac integrand and Dirac measure can hide the distinction behind a generalized BSDF, radiance or measure, where we insert either a Dirac, or a true BSDF, radiance function or measure, depending on the actual surface or light source.

4.8.2 Distributions

Let $\mathcal{C}_c^\infty(\mathbb{R}^n)$ be the set of all infinitely differentiable functions with compact support. A distribution[18] T is a continuous[19] linear function that maps such functions $\phi \in \mathcal{C}_c^\infty$ to scalar values:

$$\begin{aligned} T : \mathcal{C}_c^\infty(\mathbb{R}^n) &\to \mathbb{R} \\ \phi &\mapsto T[\phi]. \end{aligned}$$

In this context, we can view both non-singular and singular scattering as a distribution $T_{scatter}$ that maps the (non-singular) incident radiance function to the scattered radiance for some direction:

$$L_s(\mathbf{x}, \omega_o) = T_{scatter}^{\omega_o, \mathbf{x}}[L_i] = \begin{cases} L_i(\mathbf{x}, \omega_{or}) a(\omega_o) & \text{if } \mathbf{x} \text{ singular} \\ \int_{\mathbb{S}^2} L_i(\mathbf{x}, \omega_i) f_r(\omega_i, \mathbf{x}, \omega_o) \cos\theta_i \, \mathrm{d}\sigma(\omega_i) & \text{else,} \end{cases}$$

or we can view both non-singular and singular light sources as distributions that map the BSDF to the scattered radiance:

$$L_s(\mathbf{x}, \omega_o) = T_{light}^{\omega_o, \mathbf{x}}[f_r] = \begin{cases} \frac{I_p}{||\mathbf{p}-\mathbf{x}||^2} f_r(\omega_p, \mathbf{x}, \omega_o) \cos\theta_p & \text{if point light at } \mathbf{p} \\ \int_{\mathbb{S}^2} L_i(\mathbf{x}, \omega_i) f_r(\omega_i, \mathbf{x}, \omega_o) \cos\theta_i \, \mathrm{d}\sigma(\omega_i) & \text{else.} \end{cases}$$

[18] Note: Unfortunately the word "distribution" can also mean the (cumulative) distribution of random variables (section 4.2.1). In this section we only talk about distributions as mappings from functions to scalars.

[19] In this context, "continuous" means that if a sequence of functions ϕ_k converges to a function ϕ, then $T[\phi_k]$ converges to $T[\phi]$.

Just as with the integral-free representation and with singular measures, representing two singularities at once is not as straightforward. However, the context of distributions offers an alternative solution to this by expressing singularities as the limit of a series of non-singular distributions. To show this we first introduce three distribution families T_f, U_μ and ε_a, and connect them to the scattering and light distributions from above.

Distribution Families

The distribution T_f maps each function ϕ to its integral after multiplying to a continuous function $f : \mathbb{R}^n \to \mathbb{R}$:

$$T_f[\phi] := \int_{\mathbb{R}^n} f(x)\phi(x)\,\mathrm{d}\lambda^n(x).$$

The distribution U_μ maps each function ϕ to its integral with a finite measure μ:

$$U_\mu[\phi] := \int_{\mathbb{R}^n} \phi(x)\,\mathrm{d}\mu(x).$$

The Dirac distribution ε_a in particular maps each function to its value at $a \in \mathbb{R}^n$:

$$\varepsilon_a[\phi] := \phi(a).$$

In this context, the distributions for non-singular scattering and radiance belong [20] to the T_f family, with

$$T_{f_r(\mathbf{x},\omega_o)\cos}[L_i(\mathbf{x})]$$

for a BSDF f_r, and

$$T_{L_i(\mathbf{x})}[f_r(\mathbf{x},\omega_o)]$$

for incident radiance L_i. The distributions for singular lights or scattering from above are scaled Dirac distributions

$$I_p/d^2\varepsilon_{\omega_p}[f_r(\mathbf{x},\omega_o)]$$

for a point light in direction ω_p, and

$$a(\omega_o)\varepsilon_{\omega_{or}}[L_i(\mathbf{x})]$$

for a specular surface with albedo $a(\omega_o)$ and specular direction ω_{or}.

The Dirac distribution is part of the family U_μ since $\varepsilon_a = U_{\delta_a}$ with the Dirac measure δ_a. However, it is not part of the family T_f, as there exists no continuous function f such that $T_f = \varepsilon_a$. In other words, there is no function f to which we can multiply a function $\phi \in \mathcal{C}_c^\infty$ and then integrate with the Lebesgue measure to get $\phi(a)$. This is essentially

[20] Technically, T_f and distributions in general are defined for functions $\phi \in \mathcal{C}_c^\infty$, i.e. functions from $\mathbb{R}^n$ to $\mathbb{R}$. To properly express BSDF scattering or incident radiance as distributions T_f we need to include a Jacobian for transforming between solid angle integration over the unit sphere $\mathbb{S}^2$ and Lebesgue-measure integration over $\mathbb{R}^2$.

the same problem we have with mirrors or point lights in the rendering integral: There simply exist no functions that we can insert for the BSDF or incident radiance such that the rendering integral equals the remaining integrand evaluated at a single point.

Singular distributions as a limit

While ε_a is not a distribution T_f itself, it is the limit

$$\varepsilon_a = \lim_{k \to 0^+} T_{f_k}$$

of any series of distributions T_{f_k} that is parameterized by a function family

$$f_k(x) := \frac{1}{k^n} f\left(\frac{x-a}{k}\right)$$

based on a continuous normalized function f, i.e.

$$\int_{\mathbb{R}^n} f(x)\,\mathrm{d}\lambda^n(x) = 1.$$

In other words

$$\lim_{k \to 0^+} \int_{\mathbb{R}^n} f_k(x)\phi(x)\,\mathrm{d}\lambda^n(x) = \phi(a) = \varepsilon_a[\phi]. \tag{4.8.4}$$

This means we can insert a series of non-singular functions f_k for the BSDF and incident radiance in the rendering equation for a perfectly well-defined integral without any symbolic Diracs in the integrand. We can solve this integral with Monte Carlo integration, and if we take the limit $k \to 0^+$ over sampling iterations at a certain rate, we can even end up with a consistent estimate. Appendix C.1.1 shows how to do this for point lights by taking a family of shrinking disk emitters to the limit.

4.8.3 Mollification

Mollification is a form of regularization that replaces (near) singular integrands with a smoother version by performing a convolution of the (near) singular integrand with a mollifier – a family of convolution kernels where the limit of these convolutions converges to the original integrand. This can help with configurations that are hard or impossible to sample. The result is biased, but can be made consistent if the mollified integral approaches the true value at a certain rate when taking the mollifier to the limit. We apply mollification to spectral singularities in the context of fluorescence rendering in chapter 6, and to spatial (near) singularities in chapter 7.

Definition. Let $\mathcal{K} : \mathbb{R}^d \to \mathbb{R}^+$ be an infinitely differentiable function with support[21] $\mathbb{K}$ and $0 \in \mathbb{K}$. The d-dimensional mollifier based on $\mathcal{K}$ and centered at a is [108]

$$\mathcal{K}_r(x) := r^{-d}\mathcal{K}(\frac{x-a}{r}).$$

Figure 4.30 shows a one-dimensional example.

[21] Friedrichs [108] requires a compact support $\mathbb{K} = [-1, 1]$. Following the insights leading up to equation 4.8.4, mollification also works with arbitrary support, although we require $\mathbb{K}$ to include 0 for convenience.

Properties. By "centered at a" we simply mean that $\mathcal{K}_1$ is the base kernel shifted by a; we do not require the kernel to be symmetric. Since the base kernel's support includes 0, all supports $\mathbb{K}_r$ of the kernel family include a.

The functions in this family are normalized and have a compact support $\mathbb{K}_r$, with $\mathbb{K}_r \to \{a\}$ for $r \to 0^+$. Equation 4.8.4 implies that the family of distributions based on $\mathcal{K}_r$ converges to the Dirac distribution for $r \to 0^+$:

$$\lim_{r \to 0^+} T_{\mathcal{K}_r} = \varepsilon_a.$$

In other words, convolving a non-singular integrand with $\mathcal{K}_r$ converges to the original integrand in the limit. On the other hand, convolving a singular integrand in form of a Dirac distribution with a mollifier results in a non-singular distribution $T_{\mathcal{K}_r}$, whose the corresponding function we can then use in an integral. Taking $r \to 0^+$ to the limit converges back to the collapsed integral similar to equation 4.8.4.

For actual singularities, the result of a convolution of the corresponding Dirac with a mollifier is the mollifier itself. In this context we can directly replace the singularity with the mollifier.

Monte Carlo Integration and Convergence

We apply mollification in two settings: To enable Monte Carlo integration for singular equations, and to reduce noise in Monte Carlo integration for peaky integrands. Mollification introduces bias but generally reduces variance in the Monte Carlo estimator, with generally lower variance for larger kernel supports.

Kaplanyan et al. [20] show that a mollified Monte Carlo estimator for a d-dimensional integral is consistent if the base kernel support $\mathcal{K}$ is finite and the kernel scale r_n shrinks over Monte Carlo iterations n, i.e.

$$r_n < \mathcal{O}(1),$$

at a sufficiently slow rate

$$\mathcal{O}(n^{-1/d}) < r_n < \mathcal{O}(1) \tag{4.8.5}$$

such that the noise reduction that usually comes with additional samples outweighs the noise increase due to a smaller kernel support.

Previous Work

Earlier work on mollification is mainly concerned with specular connections. For instance, Kaplanyan et al. [20] regularize the *bidirectional scattering distribution function* (BSDF) with a piecewise constant mollifier. The radius is decreased iteratively to maintain a consistent estimator. They perform *multiple importance sampling* (MIS, see section 4.3.4) in the sense that the least biased strategy gets a weight of one, and they provide more theoretical background on mollification.

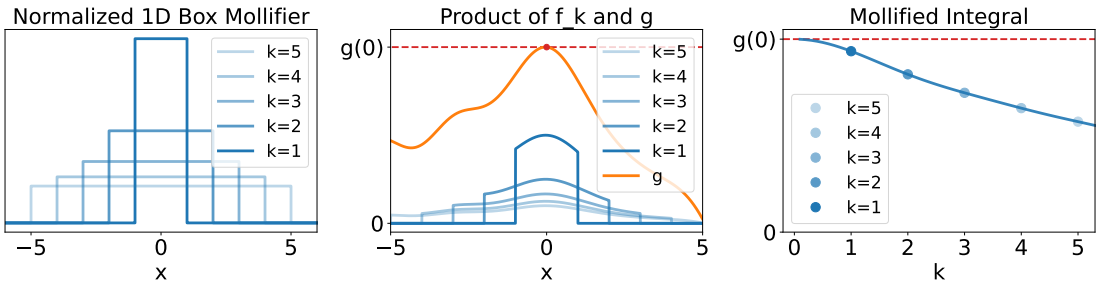


Figure 4.30: The mollifier $f_k(x) = \chi_{[-k,k]}(x)/(2k)$ (**left**) has support $[-k, k]$, and for $k \to 0$ goes to ∞ for $x = 0$ and converges to 0 otherwise. f_k is normalized, and integrating the product of f_k with some other function g (**center**) converges to $g(0)$ (**right**).

In practical scenarios, regularizing materials to aid global illumination has a long history [109]. This is often performed by changing the roughness of microfacet surfaces [107]. Bidirectional lightcuts [110] regularize the directional spread of eye sub-path tracing in a many-lights rendering context. Photon mapping and derivatives [21] [22] offer a spatial type of mollification and also employ radius shrinking techniques [35] to maintain consistency. Kaplanyan and Dachsbacher [36] present a data adaptive strategy to determine radius and shrinking rate. Outside of spatial regularization, Jarabo et al. [111] perform temporal density estimation for transient rendering with a kernel similar the one we use in chapter 6.

4.9 Evaluation of Renderer Implementations

Monte Carlo methods are the de-facto standard for solving the rendering integral to compute photorealistic images. These methods do not provide analytical solutions but only statistical estimates, and converge to either the true or biased solution with additional samples. This section introduces error metrics between two images that allow us to evaluate implementations of these algorithms by analyzing the error of their output with respect to some reference solution. In practice this has two main applications: 1) Testing if an implementation converges to a reference solution or if it is biased, and 2) evaluating the convergence rate, i.e. how the error behaves over time or sampling iterations compared to a reference implementation.

Such metrics can treat all pixels individually, using a per-pixel color difference metric, or include the overall structure of the image in an effort to better reflect how humans perceive the similarity between two images, or how intrusive certain kinds of rendering artifacts appear to a human observer.

4.9.1 Rendering Artifacts

Throughout the course of this thesis we will encounter different types of artifacts in the rendered images. All techniques in this thesis use Monte Carlo rendering, so all rendered images display some level of noise. We get both noise in terms of intensity variation, where the intensity of a rendered pixel differs from the reference intensity, and color noise, where the pixel hue varies from pixel to pixel according to the variance from the integration over the wavelength domain in spectral rendering.

Some of the methods in this thesis are biased, which means that (parts of) the images rendered with these techniques do not converge to the true solution. This manifests in form of missing image features (e.g. if a technique is unable to sample certain types of caustic paths), color bias caused by spectral regularization (chapter 6), or blurring and light leaking caused by spatial regularization (chapter 7).

4.9.2 Color Difference Metrics

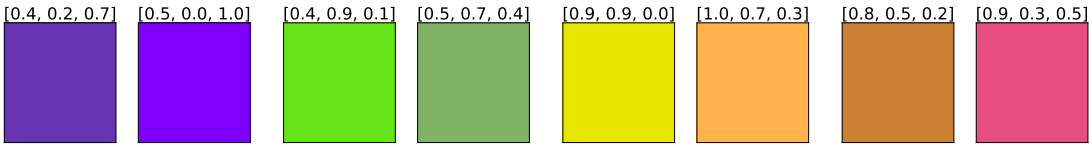
Color difference metrics judge the difference between two colors. We only consider metrics on tristimulus representations of a color. One such metric is the euclidean distance between two tristimulus colors (x_1, x_2, x_3), (y_1, y_2, y_3):

$$\Delta_{Euclid}(x, y) = \sqrt{\sum (x_i - y_i)^2}.$$

The value of this difference depends on the particular color space. For instance, two RGB color spaces with different primary colors results in a different euclidean distance for the same color pair. While equal colors have difference 0 in this metric, color pairs with the same euclidean distance do not necessarily appear equally different to a human observer. In RGB color spaces this mismatch between euclidean distance and perceived difference can be quite obvious, as illustrated by figure 4.31.

The Delta E metrics aim to better capture this perceptual difference. The Delta E 76 metric computes the difference between two colors as their euclidean distance in the CIE 1976 L*a*b color space [53]:

$$\Delta E^*_{ab}\left((L_1, a_1, b_1), (L_2, a_2, b_2)\right) = \sqrt{(L_1 - L_2)^2 + (a_1 - a_2)^2 + (b_1 - b_2)^2}.$$



ΔE^*_{ab}: 48 ΔE^*_{00}: 10 | ΔE^*_{ab}: 56 ΔE^*_{00}: 16 | ΔE^*_{ab}: 48 ΔE^*_{00}: 25 | ΔE^*_{ab}: 62 ΔE^*_{00}: 37

Figure 4.31: Four color pairs with equal euclidean distance in sRGB, and their respective difference with ΔE^*_{ab} and ΔE^*_{00}.

While this is as simple to compute as the euclidean distance in any other color space, and captures perceptual differences much better than the euclidean distance in RGB, the ΔE^*_{ab} metric is not completely uniform over all visible colors (see fig. 4.31). The ΔE^*_{2000} metric [112] improves upon the ΔE^*_{ab} metric via

$$\Delta E^*_{00}((L_1, a_1, a_2), (L_2, a_2, b_2)) = \sqrt{(\frac{\Delta L'}{k_L S_L})^2 + (\frac{\Delta C'}{k_C S_C})^2 + (\frac{\Delta H'}{k_H S_H})^2 + R_T \frac{\Delta C'}{k_C S_C} \frac{\Delta H'}{k_H S_H}},$$

where the CIELAB values (L_1, a_1, b_1) and (L_2, a_2, b_2) are combined into new terms $\Delta L', \Delta C', \Delta H', S_L, S_C, R_T$ and weighted with additional parameters k_L, k_C, k_H.

4.9.3 Image Metrics

Pixel-wise metrics. We can combine the pixel-wise color differences in any of the above metrics to form a metric for the error of the full image. The *root mean squared error* (RMSE) between an image X and a reference R is the root of the normalized sum of squared pixel differences over the entire image with N pixels:

$$RMSE = \sqrt{\frac{1}{N} \sum_N (x_i - r_i)^2}. \tag{4.9.1}$$

This error reflects the variance of the renderer, averaged over the full image. For unbiased Monte Carlo renders, the variance decreases in $\mathcal{O}(N^-1)$ (see equation 4.3.3). Since the RMSE corresponds to the standard deviation, in other words the root of the variance, it decreases in $\mathcal{O}(N^{-1/2})$. Plotting the RMSE of a series of such images over samples in a log-log-plot as in figure 4.32 (left) thus results in a roughly straight line. Vice versa, a curved line in a log-log RMSE plot as in figure 4.32 (right) indicates that the image series does not converge to the reference image with the sample count approaching infinity.

The RMSE penalizes absolute pixel differences, whereas human perception judges color difference based on hue and saturation differences, and perhaps more importantly the relative brightness difference. Typical human observers can only distinguish colors with equal hue and saturation if their relative brightness differes by at least 2%. The relative RMSE (rRMSE) alleviates the influence of pixel brightness to the error measure

$$rRMSE = \sqrt{\frac{1}{N} \sum_N \frac{(x_i - r_i)^2}{r_i^2 + \text{offset}}},$$

where we add an offset to avoid numerical instabilities for particularly dark pixels.

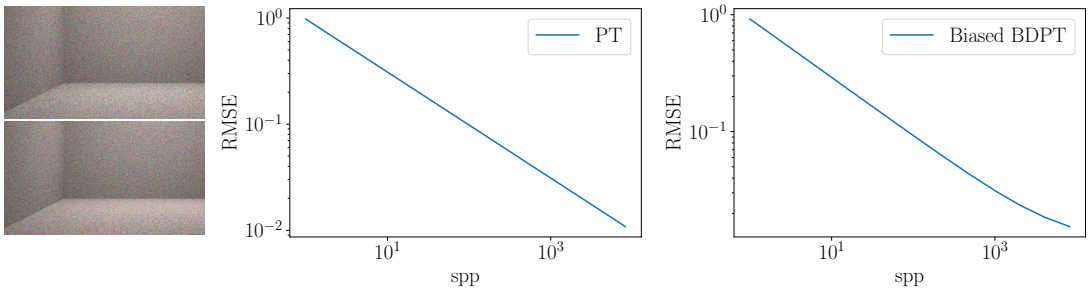


Figure 4.32: RMSE of an unbiased PT render (**top**) and an intentionally slightly biased BDPT render (**bottom**). The BDPT's bias only becomes noticeable at over 1000 samples per pixel (spp) which required two hours render time, not including the time spent rendering the reference.

Testing for bias in the form of checking whether the RMSE behaves as a straight line requires both a converged reference image, and running the tested renderer for a high enough sample count to get a reliable plot (figure 4.32).

Instead of waiting for the tested renderer to somewhat converge before computing the RMSE, Celarek et al. [113] compute many low-sample renderings and estimate the MSE's expected value and variance. This allows them to detect renderers with occasional outliers and measure the distribution of errors over frequencies. However, they still rely on a converged reference, and use the same budget for rendering many unconverged renders as would typically be used to generate one converged render.

So while RMSE plots are a good tool for comparing convergence of different renderers, they are time consuming to use for debugging purposes.

Predictive rendering. In the context of predictive rendering it is especially important that a renderer is able to match the appearance of real-world objects. Therefore renderers are validated with respect to measured data, such as measurements of real materials, light sources or photographs. One example is the Cornell Box [115] (right), a real-world model of a box, photographs of which can then be compared to the output of a renderer [116] [117]. [118] and [119] give an overview of techniques used to verify light transport compared to physical measurements for predictive rendering.



A photo of the Cornell box, source: [114]

Visual metrics. Other metrics such as [120] focus on humanly perceived image quality or similarity instead of quantitative difference to a reference render or photograph. They assign higher ratings to images that appear as plausible as possible, but do not provide any measure of the mathematical accuracy of a renderer's implementation. For instance, structural similarity (SSIM) [121] captures the effect of image artifacts such as overall bias, additional noise or blurring on a human observer better than purely quantitative errors such as RMSE.

5 Wide Gamut Spectral Upsampling with Fluorescence

This chapter is based on
A. Jung, A. Wilkie, J. Hanika, W. Jakob, and C. Dachsbacher (2019) "Wide Gamut Spectral Upsampling with Fluorescence", In: *Computer Graphics Forum* 38.4

Spectral rendering systems consider the wavelength dependency of radiometric quantities, thus offering several advantages in terms of physical realism. However, asset textures in such systems are usually drawn or acquired as RGB tristimulus values, and need to be converted to spectral data ("spectral upsampling") before they can be used for spectral rendering. While a number of RGB to spectrum upsampling techniques are available, none of them support upsampling of all colors in the full spectral locus, as it is intrinsically bigger than the gamut of physically valid reflectance spectra. But with display technology moving to increasingly wider gamuts, the ability to achieve highly saturated colors becomes an increasingly important feature.

Real materials usually exhibit smooth reflectance spectra, while computationally generated spectra become more blocky as they represent increasingly bright and saturated colors. In print media, plastic or textile design, fluorescent dyes are added to extend the boundaries of the gamut of reflectance spectra.

We follow the same approach for rendering: We provide a method which, given an input RGB tristimulus value, automatically provides a mixture of a regular, smooth reflectance spectrum plus a fluorescent part. For highly saturated input colors, the combination yields an improved reconstruction compared to what would be possible relying on a reflectance spectrum alone.

At the core of our technique is a simple parametric spectral model for reflectance, excitation, and emission that allows for compact storage and is compatible with texture mapping. The model can then be used as a fluorescent diffuse component in an existing more complex BSDF model. We also provide importance sampling routines for practical application in a path tracer.



Figure 5.1: A spectrally rendered scene with standard RGB to spectrum upsampling (**left**) cannot achieve reflectance spectra with arbitrary color saturation and brightness due to energy conservation constraints. Our technique (**center**) converts bright and deeply saturated input RGB colors to regular reflectance spectra and enhances them with a fluorescent component, reproducing colors from a considerably wider gamut. The input colors were specified in ACEScg -for instance, the top ring of the swimming pool has color (0.9, 0, 0.9) in ACEScg. To visualize colors of such high saturation, we interpret the rec2020 data of the rendered images as sRGB for display. We also provide a display-independent difference image in CIE74 ΔE (**right**), stopped down by 7ev, such that a saturated pixel value of 1.0 corresponds to $\Delta E = 128$. The squares to the left visualize the ACEScg input colors reinterpreted as sRGB.

5.1 Introduction

From a practical point of view, transitioning from traditional RGB rendering to spectral rendering brings two clear drawbacks: the first is that scene content such as textures is typically created using tools that operate in various RGB color spaces, which in turn necessitates an ambiguous conversion to spectral data before rendering can commence. This step is referred to as *spectral upsampling*.

The second issue is that energy conservation imposes a hard limit [57], *MacAdam's limit*, on reflectance color saturation in any spectral renderer: No material can elastically scatter more light than it receives at any given wavelength. This limits materials to the *gamut of solid reflectances*, which intrinsically lacks colors beyond a certain level of saturation. In some cases, however, such extreme colors are desired. This issue is not restricted to simulations. Designers of real-world materials (e.g. plastics, textiles and paint) routinely rely on optical brighteners or other fluorescent dyes to boost their overall color saturation via *inelastic* scattering.

In contrast, in an RGB-based renderer it is simple to create very highly saturated colors, especially if a wide gamut RGB space is used as "working space" in which the multiplications of albedos and incident radiance are performed. While this approach is physically questionable at best (and downright wrong for colors outside the gamut of solid reflectances), it is noteworthy that within the limits of the spectral locus, a purely RGB-based workflow does not pose constraints on how bright or saturated objects can appear.

The intersection of these two issues motivates our work: we seek to develop a convenient method for spectral upsampling of arbitrary RGB colors, even those outside of the gamut of solid reflectances. In such cases, we model the material as a combination of a regular and a fluorescent dye that jointly reproduce the input color, just as in real product design.

Our method builds on recent work for fast, minimal storage spectral upsampling [6] for the elastic component. For the fluorescent component of our model we derive a simple parametric form to describe the brighter and more saturated colors on the wide input gamut. In this work, we are mainly interested in the gamut extension capabilities offered by such an approach, and use a simple diffuse reflection model for the fluorescent component [8].

Our technique is most relevant when working with assets created in wide gamut color spaces such as DCI P3, Rec. 2020, ACEScg, etc. Spectra for smaller spaces (e.g. sRGB) can generally be created directly, without the need for added fluorescence. Our technique also enables spectral upsampling of RAW photographs that often have a much wider color gamut than sRGB.

An additional benefit of our approach is that while all of sRGB can indeed be covered by conventional reflectance spectra, the highly saturated colors near the border of the sRGB gamut require very blocky spectral shapes near MacAdam's limit of physically valid spectra. While physically plausible, such spectra are not very realistic, and in the context of Monte Carlo rendering are hard to sample. Therefore a moderate addition of fluorescence also makes sense near the boundaries of sRGB, in order to recreate the same color with a smoother reflectance spectrum.

The contributions in this chapter are:

- The first spectral upsampling technique to include fluorescence.
- A spectral model parameterized by three floating point values for the elastic component and three additional floating point values for the fluorescent part, making it small enough to be used in textured assets.
- Analytical sampling routines for wavelength importance sampling at fluorescent scattering events.

5.2 Background and Previous Work

The background chapter provides more details on color science and spectral rendering (section 4.6), fluorescence (section 4.7), and spectral uplifting (section 4.6.4) with the uplifting model by Jakob and Hanika [6] (equation 4.6.1).

Color science. Given our understanding of the human visual system (see section 4.6), color is normally expressed as a vector of tristimulus values [122]. In the case of RGB spaces, all colors that can be expressed by values in $[0, 1]^3$ form the gamut of the given color space. It has been shown that not all visible colors correspond to valid reflectances [57], due to energy conservation constraints. If one restricts oneself to natural reflectances, i.e. smooth reflectance spectra as they often appear in real materials, this gamut shrinks even more [54] (see also figure 5.2 left).

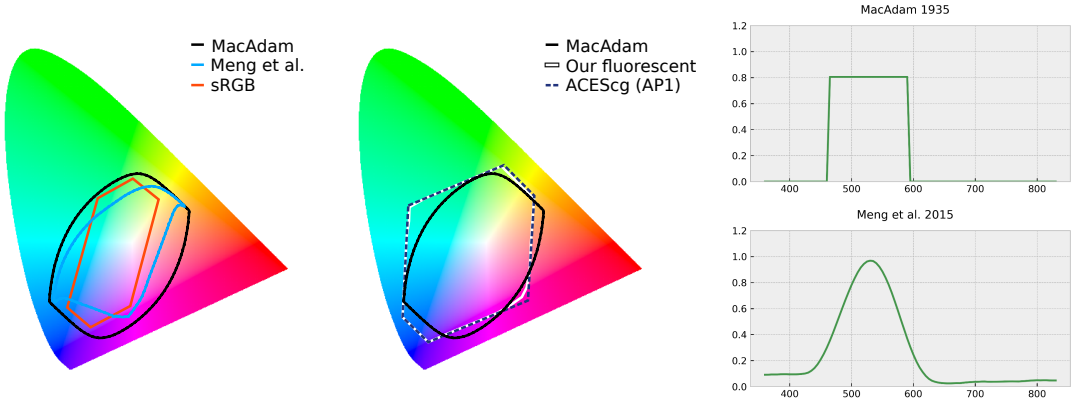


Figure 5.2: CIE chromaticity diagram showing all visible colors in a horseshoe shape, with slices through five gamuts at a brightness of $X + Y + Z = 1.5$: **Left**: The sRGB gamut (red), the limits of the gamut of valid reflectances (MacAdam's limit [123], black), and the gamut of natural (i.e. smooth [54], blue) reflectances. **Right**: The ACEScg gamut (dashed) which we use as input color space for our textures, and the approximate output color space covered by our fluorescent spectral upsampling (white). The plot on the right shows a MacAdam spectrum (**top**) and a smooth spectrum (**bottom**).

Fluorescence. For rendering we use a path tracer with next event estimation [73], together with the diffuse BBRRDF model from equation 4.7.1

$$f(\lambda_i, \omega_i, \mathbf{x}, \omega_o, \lambda_o) = \frac{\delta_{\lambda_o}(\lambda_i) \cdot (1 - ca(\lambda_i))r(\lambda_i) + ca(\lambda_i)Qe(\lambda_o)}{\pi}, \tag{5.2.1}$$

where the left summand in the numerator describes ordinary non-fluorescent reflection.

For any given input RGB color, we are now interested in providing matching $r(\lambda)$, $a(\lambda)$, $e(\lambda)$ and c. We discuss modeling a, e, c and Q in section 5.3.3. For r we rely on the sigmoid function space for spectral upsampling [6].

Outside of the upsampling context but similar to our setting, printing with fluorescent inks requires determining the fractions of regular and fluorescent ink. Rossier et al. [124] determine the extended fluorescent gamut as combination of few fluorescent primary colors in the context of half toning, given an sRGB input image. Our model specifically targets larger gamuts and uses a parametric fluorescent additive.

Spectral upsampling. As discussed in section 4.6.4, spectral upsampling approaches construct reflectance spectra to match the appearance of a given tristimulus value under certain illumination. Conventional approaches, such as the method by Jakob and Hanika [6] (see equation 4.6.1), can provide realistic reflectances that truthfully match moderate colors. Toward the boundary of the gamut of physically valid reflectances these upsampling methods produce increasingly peaky, MacAdam-like spectra.

In our work, we want to avoid such blocky spectra, but instead work with smoother spectra and mix in fluorescent dyes as optical brighteners to compensate the lack of saturation. This approach enables us to go outside the reflectance gamut and represent a richer set of colors. For this purpose, we build on the parametric function space of Jakob and Hanika [6], and we extend their set of coefficients c^r with three additional fluorescence-specific coefficients c^f that represent the peak wavelength of the emission spectrum, a mixing factor, and the fluorophore's Stokes shift.

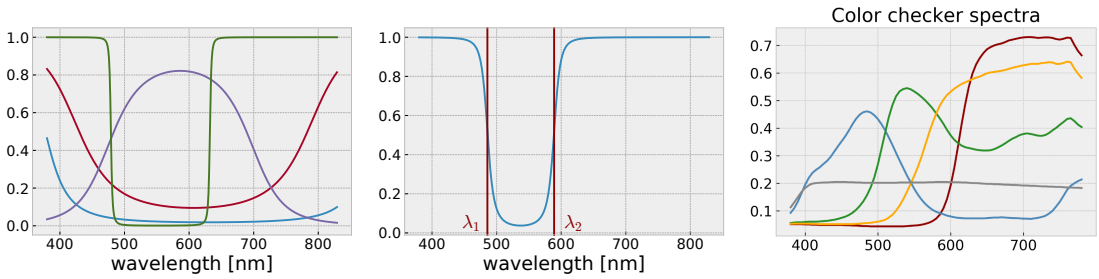
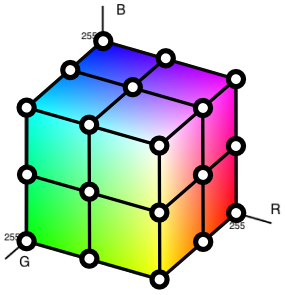


Figure 5.3: **Left**: Example spectra showing the expressiveness of the sigmoid model [6]. **Center**: Locations where we evaluate the derivative to penalize box-like spectra. **Right**: Maximum gradient of measured spectral data in the red color checker square.

5.3 Fluorescent RGB Upsampling

In this section we detail our upsampling technique. Like Jakob and Hanika [6], we compute a cube-shaped 3D lookup table, where we store precomputed coefficients ($c^r \in \mathbb{R}^3$ for reflectance, $c^f \in \mathbb{R}^3$ for fluorescence) for discrete RGB values. The three dimensions of our lookup table correspond to the three dimensions of the RGB $[0,1]^3$ cube of a given color space, such that cube entries are spaced uniformly in RGB, as in the figure on the right.



For each cube entry, we optimize the associated coefficients c^r and c^f such that we minimize the Euclidean distance inside the $[0,1]^3$ RGB cube between the target color and the color represented by the coefficients under a given illuminant. In section 5.5 we investigate alternative distance metrics.

5.3.1 Overview

We begin by populating the table for the given RGB space with coefficients c^r as computed by the spectral upsampling method by Jakob and Hanika [6]. This process does not yield satisfactory reflectance spectra for all entries when working with a wide gamut color space: Colors outside of MacAdam's gamut simply cannot be reproduced by energy-conserving spectra, while colors near the boundary to MacAdam's limit require unnaturally blocky spectra (a similar issue was reported by Meng et al. [54]). We detect the latter case using a simple analytical criterion detailed in section 5.3.2. After this, we improve the spectra computed during this first stage by determining the parameters of our analytical fluorescence model (section 5.3.3).

5.3.2 Elastic Scattering Component

Figure 5.3 shows several example reflectance spectra created via non-fluorescent spectral upsampling using equation 4.6.1. Some spectra become increasingly box-like, which is undesirable given the smoothness of natural reflectance spectra in the visible range.

To penalize such results during the optimization, we require a metric that quantifies the smoothness of the resulting spectra. We choose the highest absolute derivative value that the model $r(\lambda)$ produces on the visible range $[\lambda_{\min}, \lambda_{\max}]$ in nanometer (nm). Since an analytical computation of the true maximum absolute derivative seems not feasible, we compute an approximation instead. First, we assemble the extreme values that the function reaches on the interval:

$$\mathcal{R} \coloneqq \left\{ r(\lambda_{\min}), r(\lambda_{\max}), r(\lambda_{\text{ext}}) \right\},$$

where $\lambda_{\text{ext}} = -\frac{c_1}{2c_0}$ is the global minimum or maximum of the polynomial, which is only included if it lies between $\lambda_{\min}$ and $\lambda_{\max}$. Exploiting the monotonicity of the sigmoid S, we then map the midpoint of this range through its inverse:

$$y_{\text{mid}} = S^{-1}\left((\max \mathcal{R} + \min \mathcal{R})/2 \right).$$

To determine the associated wavelength, we must solve the quadratic equation $P(\lambda_{1,2}) = y_{\text{mid}}$. The final approximate derivative value is given by $|r'(\lambda_1)|$ or $|r'(\lambda_2)|$, either solution is suitable. An example is shown in figure 5.3.

The accuracy of this approximation increases with the maximum slope of r. For box-like spectra with values mostly close to 0 or 1, the corresponding internal polynomial P reaches extremely low and high values, which get mapped to close-to-0 and close-to-1 by the sigmoid S, with the steepest derivative of $S(P)$ at $S = (\text{almost } 0 + \text{almost } 1)/2 \approx 0.5$ for symmetry reasons. We only want to penalize spectra above a certain steepness, and do not care about smooth spectra where the approximation is less accurate.

We penalize spectra with an absolute derivative above a threshold t by adding the term $100 \cdot \max\{0, |r'(\lambda_1)| - t\}$ to the optimization objective (i.e. the distance inside the RGB $[0,1]^3$ cube). We use $t = 0.016$ (when plotted over wavelengths in nm) as a threshold for smooth spectra, which is the maximum derivative observed on the classic Macbeth color checker (see figure 5.12). We use $|r'(\lambda_1)| - t$ in order to assure a continuous optimization objective. Considering that color distances inside the $[0,1]^3$ cube can reach values in $[0, \sqrt{3}]$, we then scale this difference by a factor of 100 in order impact the target function even for slightly larger derivatives, causing the optimizer to prefer the fluorescent component to reduce color reprojection error.

5.3.3 A Parametric Model for Fluorescent Spectra

To describe fluorescence in our BBRRDF model (equation 5.2.1) we must still specify concrete absorption and emission spectra. We use a simple low-dimensional parametric model instead of densely sampled spectra to ensure that our model is efficient to use at render time. Our model can also be importance sampled analytically, which is beneficial for Monte Carlo light transport simulation. The choice of an analytic model also enables a considerably simplified optimization of model parameters for a given RGB input value.

In order to keep the number of parameters small, we fix the parameter $Q = 0.96$. We chose this value, as several real-world materials have quantum yields close to 100% [63] and we want Q to be as high as physically reasonable in order to achieve higher saturation via reradiation. Considering the model in equation 5.2.1, a lower Q would be mathematically equivalent to a larger Q coupled with lowering c and $r(\lambda)$, so we expect little benefit from introducing an additional parameter for Q.

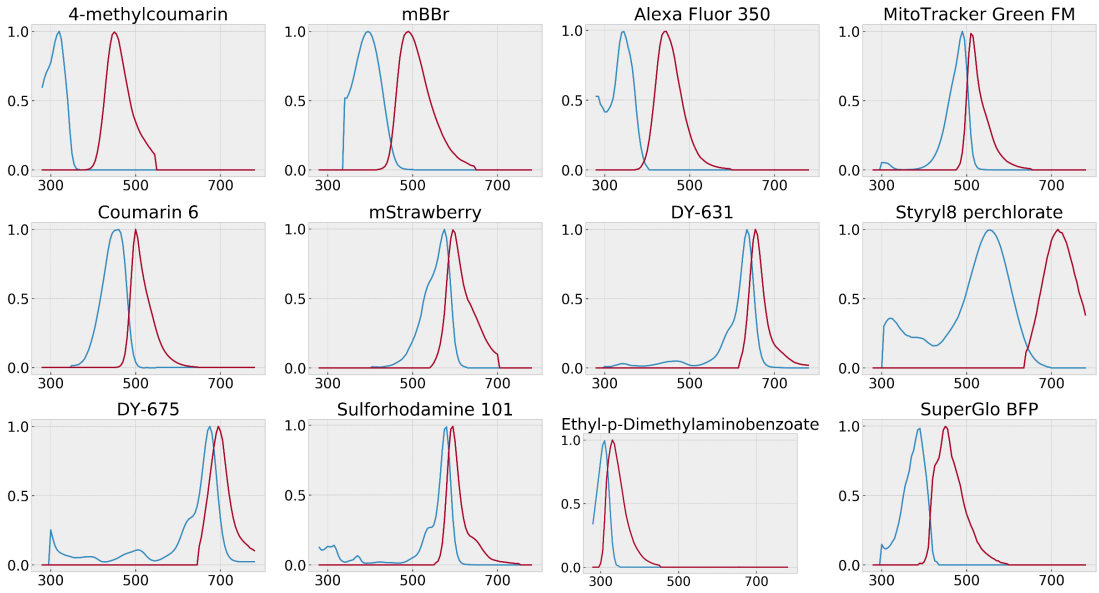


Figure 5.4: A variety of measured spectra from the Fluorophores.org database [69].

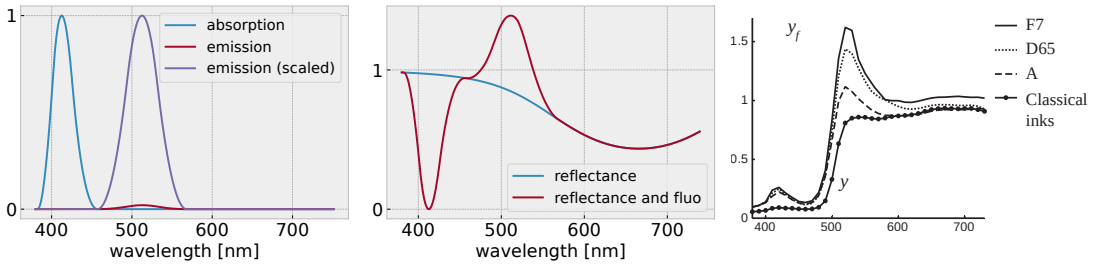
Thus our model has three parameters $c^f = (\lambda_e, c, s)$: The peak wavelength of the emission spectrum λ_e in nm, a mixing factor $c \in [0, 1]$ (as in equation 5.2.1) to determine the amount of fluorescence, and the Stokes shift s in nm, for describing the distance between the emission and absorption spectrum's peak wavelengths.

To derive reasonable parameter constraints for our model, we analyze a range of existing fluorophores in figure 5.4. We observe that most emission and many absorption spectra have a single peak and supports of roughly 100nm for the main peak. While some absorption spectra have two or even three peaks (corresponding to transitions to higher excited states S_2 or even S_3, see section 4.7), we decide to only use one peak in our model. Next, we can observe that the emission spectrum roughly mirrors the last peak of the absorption spectrum. This is called the *mirror image rule*, which states that the absorption and emission spectrum mirror each other when plotted over wavenumber; the fact that we plot over wavelength instead of wavenumber introduces a bit of a distortion. With these substances the Stokes shift (the distance between the absorption and emission spectrum peak) ranges from 15nm (Sulforhodamine 101) to 160nm (Styryl8 perchlorate). While Stokes shifts beyond 100nm are technically possible, we decide to limit the Stokes shift to 100nm to be more in line with the majority of substances.

Motivated by the similar shape of many existing fluorophores (see figure 5.4) we model the emission spectrum with a cubic b-spline

$$b(x) = \begin{cases} (x+3)^2/6 & \text{if } -3 < x \leq -1 \\ -x^2/3 + 1 & \text{if } -1 < x \leq 1 \\ (x-3)^2/6 & \text{if } 1 < x < 3 \end{cases}$$

which we stretch and translate to account for varying spectral widths and λ_e, respectively. We then define the absorption spectrum as the mirror image of the emission spectrum plotted over wavenumber, as literature suggests that this is often the case for real materials [63]. We further model each pair of emission and absorption spectra such that



Our fluorescence model | Our model's reflectivity | Real-world measurement

Figure 5.5: **Left**: In our model for fluorescent spectra the spectra mirror each other over wavenumber, so this plot over wavelengths shows the absorption spectrum as a distorted b-spline. This distortion increases with larger Stokes shift and smaller emission peak wavelength. **Center**: Apparent reflectivity of our model ($t = \infty$, no UV light, RGB=(0.52, 0.97, 0.84)). **Right**: Real life example ("daylight fluo yellow colorant"), source: [124, Fig. 1.2]. In a phenomenological comparison, both our model and the measurement exhibit a base reflectivity which stays below 1.0. Both have a fluorescent peak at the rising edge of the spectrum, which for the real material is even stronger in illumination with UV light.

they touch but do not overlap. In reality, such an overlap usually exists, but it has only a small impact on the resulting color. In our case, this assumption simplifies things because we can model the effects of excitation and emission as separable without the danger of sampling events that transition to shorter wavelengths.

We compute a scaling factor α which is used to stretch the emission spectrum, such that mirroring it over wavelength results in an absorption peak wavelength at the specified distance s as

$$\alpha = \frac{\lambda_e s}{2\lambda_e - s},$$

leading to the emission spectrum

$$e(\lambda_o) = b\left(\frac{3}{\alpha}(\lambda_o - \lambda_e)\right) \cdot \frac{9}{8\alpha},$$

and the absorption spectrum

$$a(\lambda_i) = b\left(\frac{3}{\alpha} \cdot \left(\left(\frac{2}{\lambda_e - \alpha} - \frac{1}{\lambda_i}\right)^{-1} - \lambda_e\right)\right).$$

In contrast to the absorption spectrum, which takes on the value 1 at its peak, the emission spectrum is normalized so that it integrates to 1. This normalization constant for $e(\lambda_o)$ has a closed-form solution and is given by $9/(8\alpha)$. An example of a typical resulting emission and absorption spectrum pair is given in figure 5.5.

Spectral comparison. We compare the phenomenological appearance of the reflectivity created by our model with the reflectivity of a real-world material that was enhanced with a fluorescent dye in figure 5.5. Both spectra are mostly below 100% reflectivity except for the fluorescent peak. The synthetic and real fluorescent peak are of roughly similar shape, amplitude and width.

5.3.4 Parameter Search

Target function. We minimize an objective function given by the Euclidean distance between the input color and the color represented by the coefficients. To compute the latter we evaluate the reflectivity R for a given illuminant $I(\lambda)$ as

$$R(\lambda) = r(\lambda) \cdot \big(1 - ca(\lambda)\big) \cdot I(\lambda) + e(\lambda) \cdot \int_{-\infty}^{\infty} ca(\bar{\lambda}) I(\bar{\lambda}) \, \mathrm{d}\bar{\lambda},$$

and convert it to RGB using the color matching functions associated with the desired color space. If no fluorescence is present, the above expression reduces to computing the color of the reflectance spectrum r under a given illuminant I. We furthermore add a penalty for large derivatives as described in section 5.3.2.

Algorithm. An outline of our approach is given in algorithm 2. For the optimization we use the Ceres solver [125], both for the initial search for reflectance coefficients c^r, and for finding fluorescent coefficients c^f wherever reflectance alone does not lead to a satisfactory result. This can happen when there is no physically valid reflectance for the given input color, or when a physically valid reflectance exists but is rejected for being too box-like.

In the initial search, Ceres finds zero-error c^r for colors inside the gamut of smooth spectra. In the following we only consider entries where those coefficients still produce some remaining error. We will discuss finding fluorescence coefficients c^f wherever required, and refining c^r in combination with c^f wherever c^r alone is not enough.

We found that whenever fluorescence is required, the performance of the optimizer heavily depends on the quality of the input coefficients, and it easily gets stuck in local minima. We therefore support the optimizer in two ways: Initially, we use a priori knowledge about the fluorescence parameter space to find a "good" initialization. Next, we iterate multiple calls to Ceres, while propagating information from neighboring cube entries. We always cache the best coefficients found so far, in case one of these steps increases error.

Initialization. After the initial search for c^r, we need to find a good initial guess for c^f to guide Ceres. We already know the meaning (emission peak wavelength, concentration, Stokes shift) of the fluorescence coefficients, which leads us to reasonable bounds for their value range (details in section 5.4). We then use these bounds to perform a brute-force search over the three-dimensional fluorescent coefficient space to find initial values with low error, given the initial c^r. Figure 5.6 visualizes a one-dimensional slice of such a brute-force search for fixed c^r, c and s, revealing an intricate optimization objective with several local minima.

Iterative refinement. We found it most effective to call Ceres multiple times, alternately optimizing both c^r and c^f jointly, and only optimizing c^r (algorithm 2). If smooth reflectances are desired, the target function inside these calls penalizes large derivatives as described in section 5.3.2. We terminate this process after a maximum number of iterations, or when the error no longer improves.

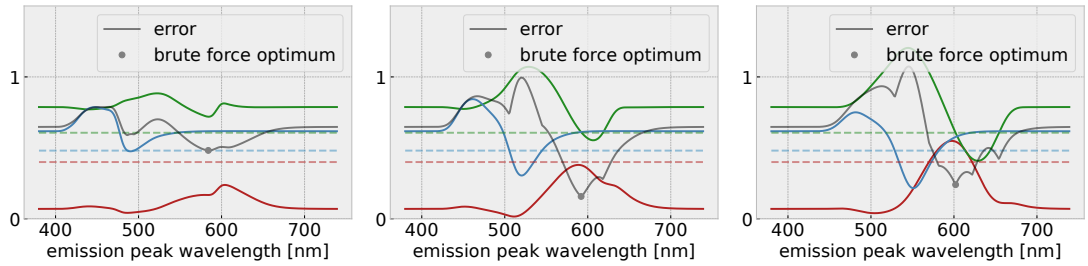
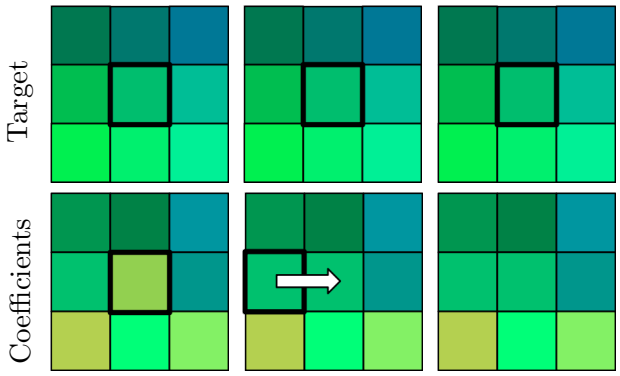


Figure 5.6: One-dimensional brute-force search for a low-error emission peak wavelength λ_e, given a fixed Stokes shift (**left to right**: $30, 60, 90$nm), mixing factor ($c = 1.0$) and reflectance coefficients. The goal is to match RGB=$(0, 0.80, 0.58)$ (dotted lines). The colored lines represent the achieved color depending on λ_e. The gray line is the RGB-distance between goal color and the color represented by the coefficients. The gray dot marks the best value found with a step size of 2nm.



for *Entries* e **do**
  Δe = e.target - e.coeffRGB
  if $\Delta e > \varepsilon$ **then**
    for *Neighbors* n **do**
      Δn = e.target - n.coeffRGB
      if $\Delta e > \Delta n$ **then**
        e.coeff = n.coeff

Figure 5.7: For each entry where the coefficient RGB is too different from the target RGB, we iterate the entry's neighbors. If any of the neighbors' coefficients is a better match for the center entry's target RGB, we copy over the neighbor's coefficients as seed for the next iteration.

Information from neighbors. While the procedure described so far finds optimal solutions for some RGB values, for many others it gets stuck in local minima. To improve convergence globally, we propagate solutions from adjacent cube entries (see figure 5.7): for each entry, whose coefficients c_j^r, c_j^f have not yet converged below some error threshold, we look at the coefficients c_k^r, c_k^f of adjacent entries $k = 1, \ldots, 26$. Coefficients that would result in a lower error for the center entry than c_j^r, c_j^f are then used to initialize the starting guess of the subsequent iteration.

We then resume optimizing c^r and c^f jointly and optimizing c^r only, reseeding using neighbor information, and so on. This process repeats a number of times to ensure that information can propagate to more distant entries.

We found that results further improve when decreasing the error thresholds over iterations (implemented by comparing neighboring error to an upscaled center error), and also starting from the entry's own coefficients if no better neighbors are available; hence, the loop over k in algorithm 2 begins at $k = 0$ to represent the center entry j.

Algorithm 2: Parameter Optimization

```
forall Cube entries do
    optimize c^r
forall Cube entries with remaining error do
    brute-force search good initialization for c^f
for i = 1, ..., 50 do
    forall Cube entries j with remaining error do
        forall k = 0, ..., 26 neighbors with error ≤ i· own error do
            c^r_j ← c^r_k, c^f_j ← c^f_k
            for l = 1, ..., 31 do
                if c^r_j, c^f_j still have error then
                    if l mod 2 then
                        optimize c^r and c^f jointly
                    else
                        only optimize c^r
```

5.3.5 Importance Sampling

Our model admits a simple spectral importance sampling technique for use in Monte Carlo rendering systems. Given a material interaction, we must first decide whether to scatter elastically or inelastically, for which we use the probabilities specified in [8]. In case of inelastic scattering, the next step involves importance sampling a new wavelength from the absorption spectrum $a(\lambda_i)$ on a camera path, or from the emission spectrum $e(\lambda_o)$ on a light path.

We can sample $e(\lambda_o)$ precisely using the following expression:

$$\lambda_\mathrm{o} = \lambda_e + \alpha \left(\frac{2}{3}(\xi_1 + \xi_2 + \xi_3) - 1 \right).$$

$\xi_1, \ldots, \xi_3$ are uniformly distributed random numbers on the interval $[0, 1]$. This relies on an idea by Stark et al. [126], which exploits the convolutional nature of b-splines: any b-spline can be expressed as a repeated convolution of box functions, which can hence be sampled by summing a suitable number of uniformly distributed variates.

The same approach unfortunately cannot directly be applied the absorption spectrum $a(\lambda)$, since the underlying spline curve is mapped through a nonlinear transformation. We currently sample a triangular density function, whose peak and support match those of $a(\lambda)$, though we note this scheme could be significantly improved e.g. by sampling a normal distribution or suitably parameterized b-spline. The integral of the absorption spectrum required for computing the probability of a fluorescent event on a camera path as specified in [8] can be approximated with the integral of a b-spline with the same support.

In the elastic case, we simply preserve the current wavelength and multiply the path throughput by the reflectance spectrum $r(\lambda)$ divided by the probability for elastic scattering.

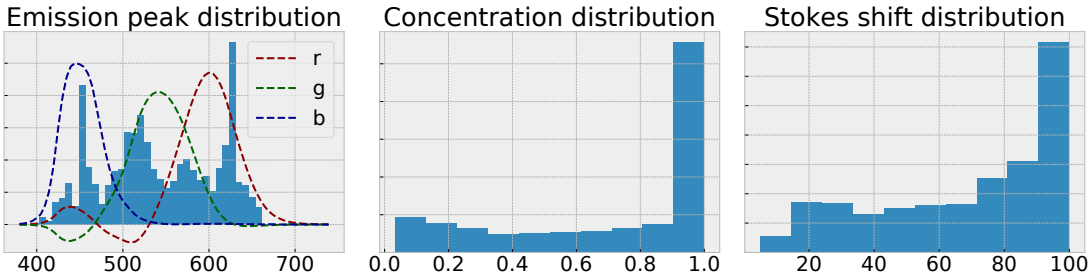


Figure 5.8: Distribution of λ_e, c and s for entries using fluorescence with $t = \infty$ and no UV light. The ACEScg RGB response curves are overlaid on the emission peak distribution.

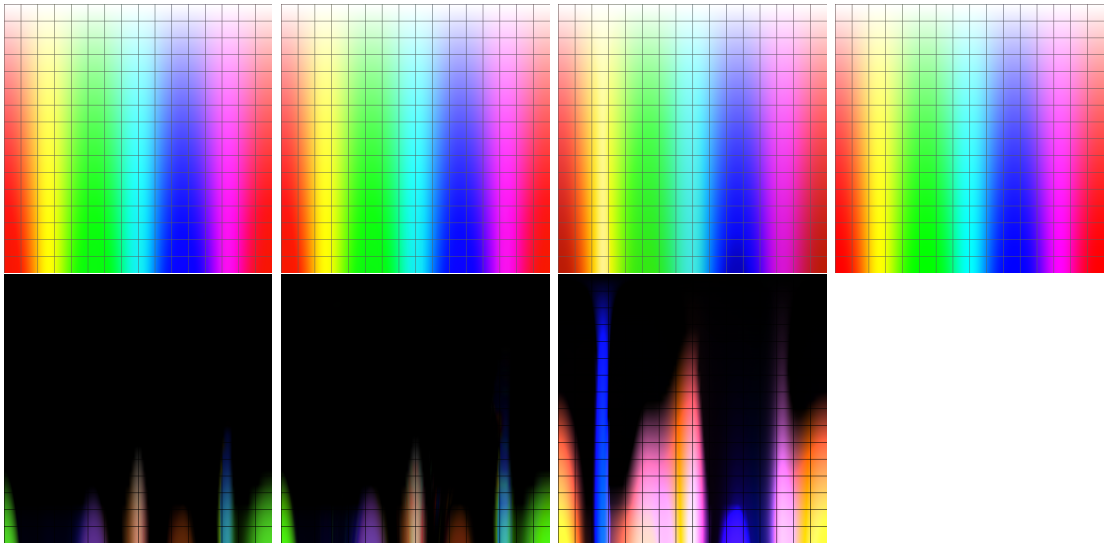
5.4 Results

We implemented the simple diffuse fluorescent BBRRDF from equation 5.2.1 in the context of a standard path tracer with next event estimation in an existing spectral rendering framework. All our experiments were run on an Intel i7-6700 3.4GHz CPU.

How to interpret the colors in this section. All figures in this thesis show an sRGB interpretation of RGB triples from some potentially different color space. The figures in this section in particular (i.e. the cube layers, the rainbow textures, and the gamut visualizations) got ACEScg RGB triples as input, ignore that these triples are supposed to be interpreted with respect to the ACEScg primaries, and instead pretend that they are triples in sRGB and show them as such. This means that what we see in the following figures are not the correct colors but an sRGB re-interpretation that warps the ACEScg gamut to the sRGB gamut.

Optimization parameters. We tested our optimization for the ACEScg color space. The cube entries are placed in a regular grid at $[k \cdot 1/32]^3$, $k = 0, .., 32$. After the initial search for c^r, the brute-force search evaluates all combinations of emission peak wavelengths in $[300, 800]$ in 10nm-steps, Stokes shifts in $[5, 100]$ in 10nm-steps, and mixing factors in $[0, 1]$ in 0.1nm steps to initialize c^f wherever required. While a search with smaller step sizes (e.g. 1nm for emission peak and Stokes shift) initially yields lower-error coefficients, it takes considerably more time (46min for 1nm steps versus 3min for 10nm steps) and the initial benefit vanishes after several rounds of seeding with adjacent coefficients. We performed 50 iterations considering adjacent coefficients as seed, each followed by up to 31 calls to Ceres, alternating joint optimization of c^r and c^f, and optimization of c^r only.

Cube statistics. For $t = \infty$ (i.e. arbitrarily large reflectance derivatives), both with and without UV light in the illuminant, the whole optimization took 2h 43min. In this case 5728 entries required fluorescence, 1486 of which still have non-negligible error when the optimization is complete. For $t = 0.016$ this number increases to 1908, with a total of 16333 entries requiring fluorescence, leading the optimization to last 4 hours. Figure 5.11 compares the final error for $t = 0$ and $t = 0.016$.



fluo, $t = \infty$ | fluo, $t = 0.016$ | no fluo | reference

Figure 5.9: Texture reconstruction comparison. **Left to right**: Fluorescence and $t = \infty$, fluorescence and $t = 0.016$, no fluorescence, input reference. The image values are meant for ACEScg but are displayed as sRGB. The bottom row shows 10 times the absolute difference in RGB. RMSE: no fluo: 0.10443, fluo ($t = \infty$): 0.01549.

We visualize the distribution of optimized λ_e, c and s in figure 5.8 for all entries with $c > 0$, i.e. all entries that actually use fluorescence, for the $t = \infty$ cube. The emission peaks are somewhat correlated to the color matching functions' peaks at roughly 450, 530 and 630nm. The concentration and Stokes shift histograms show that larger values are preferred, which makes sense as those increase the amount of reradiated energy.

Jointly optimizing c^r and c^f in all Ceres calls takes twice as long to achieve the same quality as alternating joint optimization of c^r and c^f, and isolated optimization of c^r. Alternating isolated optimization of either c^r or c^f produces a worse result, where 2782 entries still have a non-negligible error after almost 4 hours for $t = \infty$ without UV light.

HSV test card. In a first experiment, we directly convert texels of a 2D ACEScg texture (figure 5.9 right) to colors that can be represented by our model. To do so, we perform lookups into the precomputed coefficient cube and use trilinear interpolation to interpolate reflectivities of the cube's entries. Next, we evaluate the resulting RGB reflectance using the same illuminant that was also used to create the coefficient cube.

We compare the error of the input texture to the resulting color for a cube without fluorescence (i.e. only optimizing reflectance coefficients), and two cubes with fluorescence in figure 5.9. In order to show what is theoretically possible we compare arbitrary derivatives with and without fluorescence, and a derivative threshold of $t = 0.016$ with fluorescence. Both the input texture and the reconstructed textures store ACEScg tristimulus values, which on screen are re-interpreted as sRGB, thus visualizing a desaturated version of the true colors. The addition of fluorescence increases the number of colors that can be reconstructed with zero error, and reduces the error for the remaining colors. We could not achieve zero error on the full ACEScg gamut, which is most obvious in the dark blue and red regions.

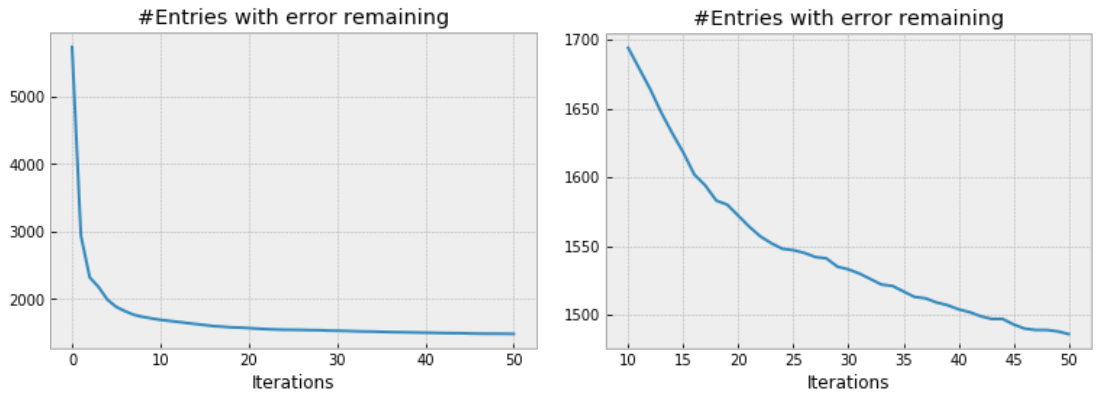
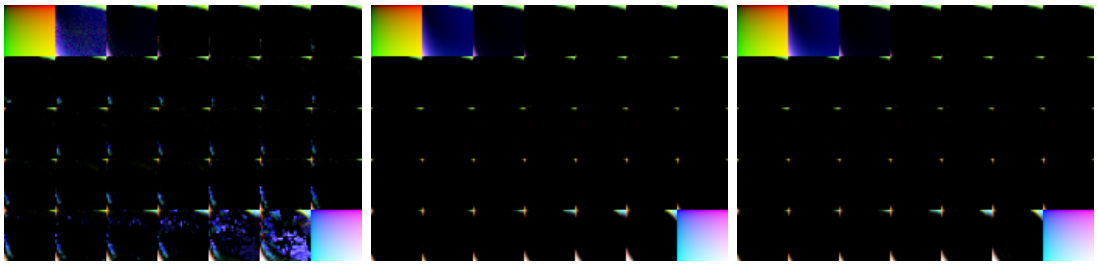


Figure 5.10: Remaining entries with error above our error threshold. The brute force search and neighbor iteration drastically reduce the error in initial iterations (left), though subsequent iterations still show non-negligible improvements (right).



$t = 0.016$, no UV light $t = 0$, no UV light $t = 0$, UV light enabled

Figure 5.11: Overexposed absolute error of individual cube entries at 1/32 slices throughout the cube. The upper left and lower right square show the true color of the first and last slice of the cube. The blue axis increases from slice to slice, the red (green) axis from left to right (top to bottom) within each slice. At the borders and first and last slice, the corresponding channels are 0 or 1.

Cube entry errors. Figure 5.10 visualizes the convergence speed of the optimization by showing how many entries above a certain error threshold remain after each iteration. Figure 5.11 visualizes the component-wise absolute error of the color for each entry of the coefficient cube. The black region in the center corresponds to the colors that can be reproduced perfectly by either reflectance only or reflectance plus fluorescence. The first layer corresponds to the blue values of 0, which poses the biggest issue for the optimizer. The upper row and left column correspond to green and red values of 0, respectively.

Gamut. We evaluate the approximate gamut of our fluorescent upsampling model in figure 5.2 (right). This figure has been drawn based on data assuming a D65 illuminant, and results may vary for other illuminants. It shows that the fluorescent upsampling model is able to reach outside MacAdam's limit, and almost fills all corners of the wide ACEScg gamut. There are some slight differences remaining in cyan and more obvious gaps in the magenta corner. As our search space is limited by ACEScg, we do not reach full coverage of MacAdam's limit. To evaluate the limits of our method, we would need an even larger input color space.

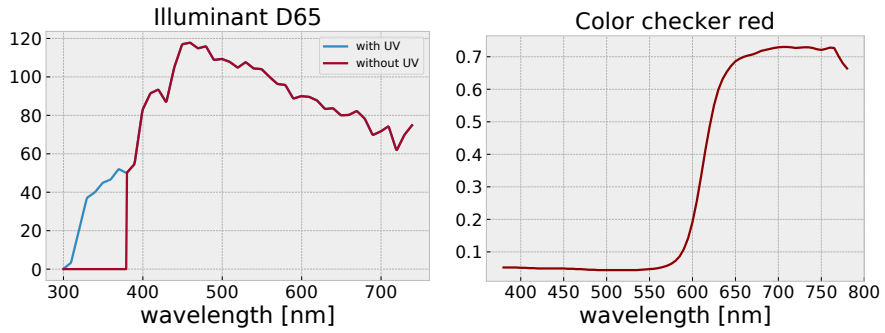
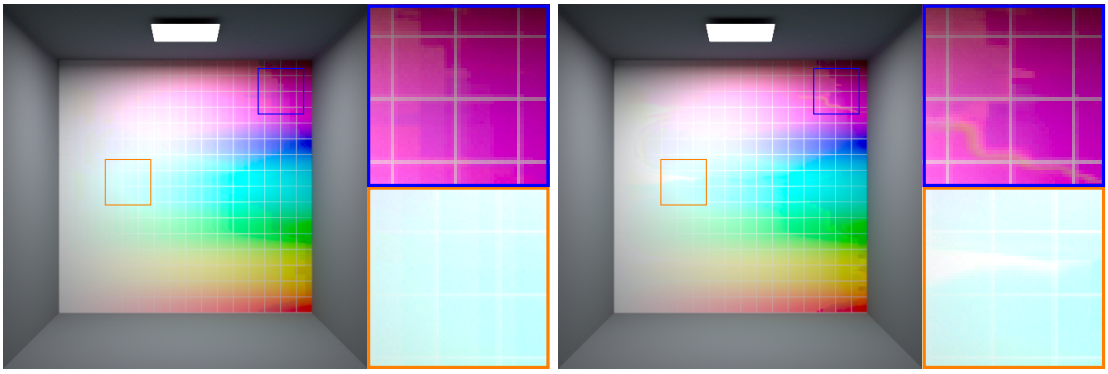


Figure 5.12: **Left**: Illuminant D65 with and without an ultraviolet component. **Right**: Color checker red reflectance spectrum.



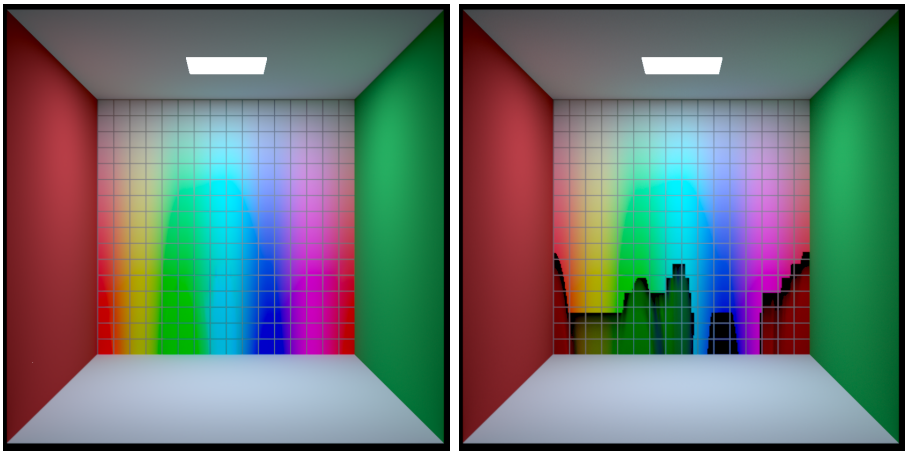
Nearest coefficients Interpolated coefficients

Figure 5.13: A colorful input texture converted to coefficient textures using the nearest (**left**) and interpolated (**right**) coefficients. Using nearest coefficients produces many seams with low contrast. Interpolation results in a mostly smooth image, but with few, more disruptive artifacts. These are most noticeable along the transition to where fluorescence is required, but can also be spotted in the white regions near the left wall.

UV light. Figure 5.11 reveals that adding ultraviolet light to the illuminant as in figure 5.12 does not decrease the overall error. This might be partly because the ultraviolet component of D65 is much weaker than the visible component. Another possible reason is that due to the limited Stokes shift, adding UV light would mostly influence bluish colors. However, figure 5.11 shows that colors with large blue and small red and green values inside the cube can already be realized well without fluorescence.

Reradiation matrices. The reradiation matrix corresponding to the spectra can be created by evaluating the BRDF in discrete steps. Examples of such matrices are presented in figure 5.15. Since the absorption and emission spectrum do not overlap, the area corresponding to fluorescence is entirely above the diagonal.

Cube interpolation. We evaluate three different strategies for converting an RGB texture to a coefficient texture. We always interpret the texel's RGB value as cube coordinates in $[0, 1]^3$. We compare using coefficients from the coefficient cube's nearest entry to interpolating coefficients of the 8 surrounding cube entries in figure 5.13.



Interpolate reradiation matrices Reflectance = 0 if fluorescent

Figure 5.14: **Left**: Here we evaluate all 8 reradiation matrices involved in a given coefficient cube voxel, and analytically sample the discrete reradiation matrix defined by the interpolated coefficients. While this requires a lot of storage (we have to explicitly generate the entire reradiation matrix for each texel), this is the proper way to do it, and leads to a smooth result. **Right:** This image has reflectance set to 0 wherever fluorescence is present, and shows how fluorescence increases as chroma goes up. Some hues, like e.g. orange, do not seem to require fluorescence as much as others, as the gamut of reflectance spectra reaches further in some regions (see figure 5.2).

As both these strategies come with drawbacks, we also present a rendering based on true interpolation of the reradiation matrices represented by the 8 surrounding cube entries in figure 5.14. Strictly speaking this does not result in a low-storage texture representation anymore, since each texel now stores a full reradiation matrix, which is why this figure uses a lower resolution input texture. Examples for interpolated reradiation matrices are presented in figure 5.15. A material corresponding to such an interpolated matrix would contain several fluorophores mixed together, while at discrete cube entries the corresponding material contains only one fluorophore.

When interpolating fluorescence coefficients we have to take care to only interpolate emission peaks λ_e and Stokes shifts s of entries that actually incorporate fluorescence, i.e. entries with $c > 0$. Otherwise the values stored for λ_e and s are meaningless. The mixing coefficient c itself needs to be interpolated with regular interpolation weights.

It turns out that when transitioning from a region that does not require fluorescence to a region that does, the reflectance spectra's brightness drops significantly. Due to the nonlinearity of the sigmoid mapping, a trilinear interpolation of reflectance coefficients does not cause a linear drop of the reflectance spectrum's intensity. Meanwhile, the fluorescent component's intensity does behave linearly with the interpolation of c. This combination of an overly bright interpolated reflectance spectra with the linearly interpolated fluorescent brightness causes a glow-like effect in the transition region. In conclusion, while interpolating coefficients produces reasonable results for smooth regions inside the cube, it is not a good idea in general due to the abrupt change in reflectance once fluorescence is incorporated. On the other hand, using the nearest entry's coefficients results in a larger number of hard transitions, albeit less noticeable ones.

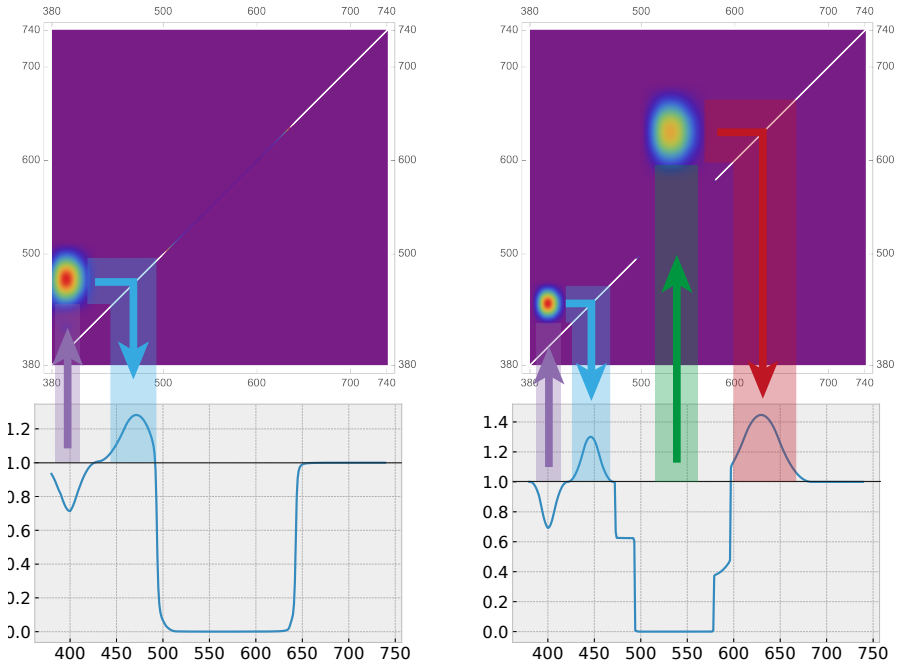


Figure 5.15: Reradiation matrices with input from all 8 voxel corners (**top**), and corresponding reflectance spectra under illuminant E (**bottom**). The example on the left is typical: The 8 corners of the voxel for a particular RGB value place several almost identical reradiation lobes in close proximity, so that the resulting overall reradiation is still confined to a single area in the matrix. In this sort of scenario coefficient interpolation yields reasonable results (reflectance brightness issues aside). But sometimes, as in the scenario on the right, two or more clearly distinct reradiation components corresponding to different emission colors necessitate a full matrix reconstruction.

5.5 Extensions

This section is based on
L. König, A. Jung, and C. Dachsbacher (2020) "Improving Spectral Upsampling with Fluorescence", In: *Workshop on Material Appearance Modeling* (R. Klein and H. Rushmeier, eds.), The Eurographics Association

In a follow-up work we introduced three extensions to this approach:

- We propose an alternative parameterization for the sigmoid function space of non-fluorescent reflectance spectra with more intuitive parameters and better behaved interpolation.
- We replace the analytical fluorescence model with a set of real-world fluorescence measurements in order to accurately reproduce appearance with specific dyes.
- We analyze how the color error metric affects the optimization process and demonstrate faster convergence with the ΔE_{76} and ΔE_{00} metrics.

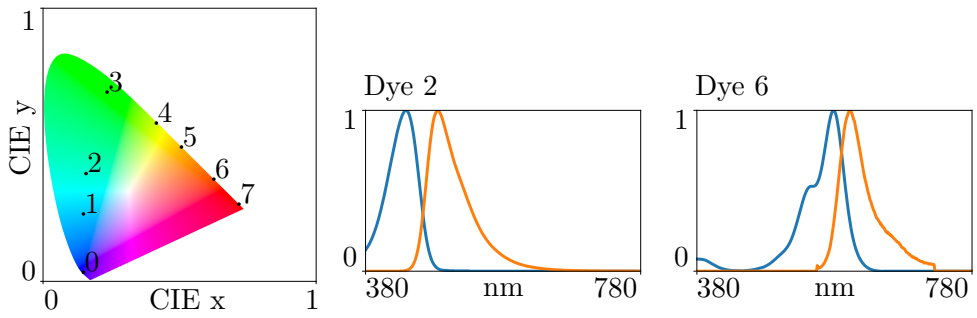


Figure 5.16: **Left**: Chromaticity of 8 fluorescent dyes without non-fluorescent reflection. **Right**: Absorption (blue) and emission (orange) spectra of dyes 2 and 6.

5.5.1 Perception Based Color Distance Metric

In order to optimize parameters (e.g. $[c_0, c_1, c_2]$ in [6] or $[\lambda_e, s, c]$ in section 5.3.3) to minimize the difference between a target color and the color represented by them, we need a meaningful metric for this difference. The original approach uses the Euclidean distance in RGB space, which approaches zero as the colors become more similar, but ignores the perceived error.

To test if other metrics offer room for improvement, we employ the ΔE^*_{76} and ΔE^*_{00} color difference metrics, This aids the optimization in convergence for colors that can be represented perfectly, and in finding perceptually better approximations for colors that cannot be represented by the analytical model.

5.5.2 Fluorescent Dyes

For a more realistic fluroescence model we replace the analytical model from section 5.3.3 with measurements from fluorescent substances. We chose eight fluorescent dyes from online databases (dyes 1 and 2: [127], others: [91]) with absorption peak wavelengths ranging from 390nm to 633nm and sort them based on their emission spectrum's chromaticity as shown in figure 5.16.

Our model combines two adjacent dyes with an interpolation weight $w \in [0, 1]$ to achieve a greater range of colors. Just as with the analytical approach, we need three coefficients to model fluorescence: The mixing coefficient c, interpolation weight w and index idx of the first dye. If memory is an issue this model may be easier to compress as idx can only take on eight values. For simplicity we interpolate the absorption and emission spectra of the two dyes. Thus the combined absorbed energy is reradiated according to the combined emission spectra. It would be more accurate to calculate absorption and emission for both dyes separately and interpolate the result.

This model affects the optimization process, since the index idx can only take on discrete values. We use a brute-force iteration over all eight pairs of adjacent dyes and optimize the remaining coefficients to find the best fitting pair and coefficients.

Another challenge is interpolation between colors, as adjacent entries in the precomputed coefficient cube can have different dye indices. A possible solution would be a BRDF that includes all fluorescent dyes referenced by the coefficients of all corners of a cube grid cell. Using this approach would allow to use only three fluorescence coefficients per grid point, while still being able to interpolate them.

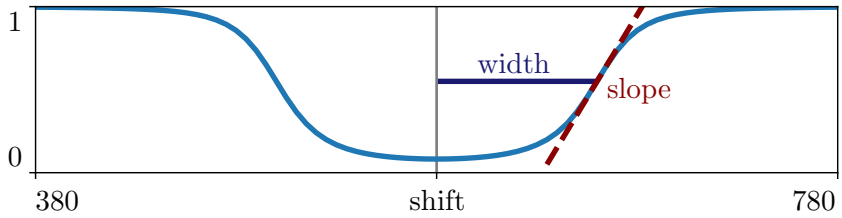


Figure 5.17: Parabola features *shift*, *width* and *slope*.

5.5.3 Alternative Interpolation

As it is infeasible to precompute coefficients for every single color in an RGB color space, we need to interpolate the upsampling model inbetween the precomputed grid points. It is thus important to define coefficients in a way that allows for a meaningful interpolation, with a result as close as possible to a true interpolation of the corresponding spectra.

This is particularly important at the border between coefficients for spectra with and without fluorescence, where the reflectance spectra differ more. The reflectance model (equation 4.6.1) is highly non-linear for linearly changing coefficients c_0, c_1, c_2. We present a new parameterization for this function space to improve interpolation in such cases (Figure 5.18).

We use three features of a reflectance spectrum as coefficients: the *shift* (i.e. the center wavelength), the *maximum slope* and the *width* of the parabola measured between the maximum slope and shift as shown in figure 5.17. Note that the maximum slope value is defined as the maximum absolute slope of a parabola, but *slope* itself is positive or negative to indicate a u-shaped or n-shaped parabola respectively.

These coefficients are related to certain color features. The shift corresponds to the chromaticity. The width and the slope simultaneously affect saturation and brightness, which corresponds to the fact that physically valid reflectance spectra cannot appear arbitrarily bright *and* saturated. The color's brightness also depends on the shift, as the human eye is less sensitive to wavelengths near the border of the visible range.

To create a spectrum from the features *shift*, *width* and *slope* (z, w and s), we use algorithm 3 to transform them to coefficients (c_0, c_1, c_2) before evaluating equation 4.6.1.

Algorithm 3: (*shift* z, *width* w, *slope* s) $\rightarrow (c_0, c_1, c_2)$

$t \leftarrow (|s| \cdot w + (s^2 \cdot w^2 + 1/9)^{1/2})/(2 \cdot |s| \cdot w)$
$c_0 \leftarrow s \cdot t^{3/2}/w$
$c_1 \leftarrow -2 \cdot c_0 \cdot z$
$c_2 \leftarrow c_0 \cdot z^2 + s \cdot w \cdot (5 \cdot t^{3/2} - 6 \cdot t^{1/2})$

Challenges. This model struggles with representing unsaturated, gray colors. We can specify colors that are perceived as gray by creating an almost horizontal line using a large width and small slope. However, a given gray color can in fact be represented by infinitely many combinations of *width* and inversely scaled *slope* values, while varying the $shift$ causes little noticeable effect. This can cause artifacts when interpolating between these extreme values (see figure 5.23).

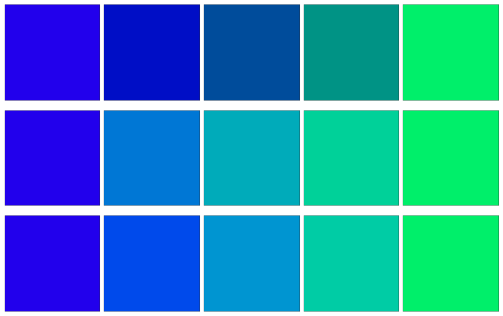


Figure 5.18: Linear interpolation of the parabola coefficients (**top**), the color spectrum itself (**center**) and the feature based coefficients (**bottom**) with weights 0, 0.25, 0.5, 0.75, and 1.



Figure 5.19: Part of the *blue* = 0.7 layer of the coefficient cube. **Left**: Based on the Euclidean distance in the RGB color space, 32 iterations. **Right**: Based on the ΔE^*_{00} color distance metric, 4 iterations.

Another problem is the distinction between u-shaped and n-shaped spectra, which are caused by positive and negative *slope* values respectively, as spectra with slope values near 0 behave differently depending on the sign. For typical widths, small positive slopes produce bright colors, while small negative slopes produces dark colors. This causes artifacts when interpolating from a u-shaped to an n-shaped parabola.

We avoid this issue with a hybrid interpolation approach, where we transform our feature based coefficients to the original ones (c_0, c_1, c_2) before interpolating such cases.

5.5.4 Results

In this section we visualize layers of an optimized 32^3-coefficient grid in the ACEScg color space. We use the ACES primaries #1 (AP1) for optimization and analysis, but interpret the resulting triples as sRGB for visualization. Reflectance spectra are limited to smooth spectra with a maximum slope of 0.01. UV light is included for the optimization of the fluorescent coefficients. The cube grid layer visualizations represent a layer of the cube grid for a fixed value of the blue component. We use the CIE D65 illuminant as light source and the CIE 1931 2 degree standard observer functions. For visualization purposes we plot fluorescent emission spectra (Figure 5.16, orange) scaled to 1.

Perception based color difference metric. The ΔE^*_{76} color difference metric was easy to implement and leads to a significantly smoother color transition through the coefficient cube as shown in figures 5.19 and 5.20, but also introduces artifacts in the red/green part of the coefficient cube when used with too few iterations. Using the computation heavier ΔE^*_{00} color difference metric or increasing the number of iterations which propagate the coefficients to adjacent grid points in the coefficient cube removes the artifacts as shown in figure 5.21.

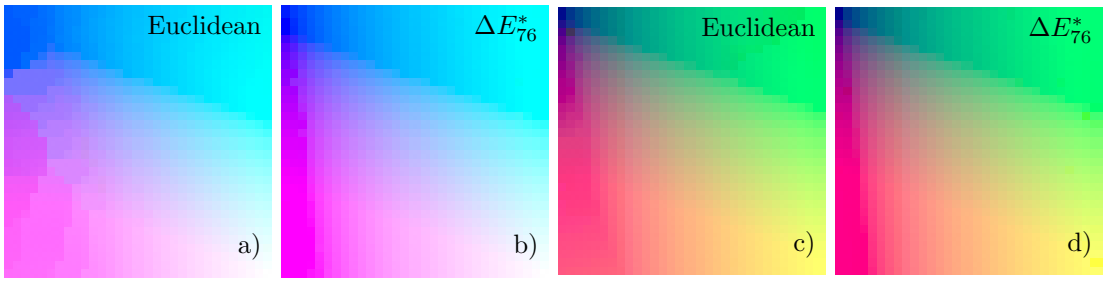


Figure 5.20: Cube layers 31 ($B = 1$; a, b) and 8 ($B = 0.258$; c, d) for the Euclidean distance in RGB (a, c) and the ΔE^*_{76} metric (b, d), both with 4 iterations.

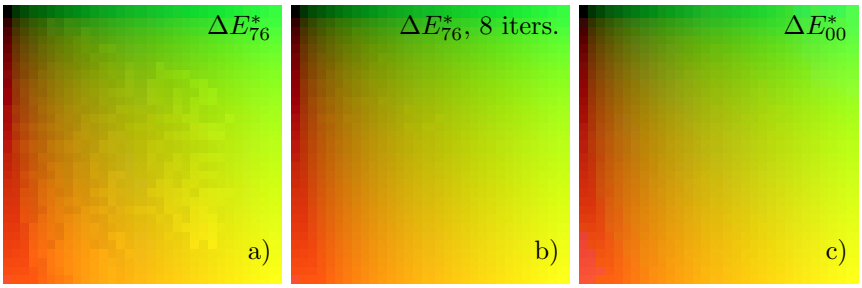


Figure 5.21: Cube layer 0 ($B = 0$) for the ΔE^*_{76} metric and 4 iterations (a) has artifacts in the red/green region, which mostly disappear with 8 iterations (b) or the ΔE^*_{00} metric (c).

Although both ΔE^* metrics require an additional step to convert the color from the initial RGB color space to the CIE L*a*b* color space, overall convergence of the optimization was faster, decreasing from 5h 59m (Euclidean distance in RGB) to 3h 18m (ΔE^*_{76}) for otherwise equal parameters and 4 iterations. Figure 5.21 shows that remaining artifacts are removed best by switching to the ΔE^*_{00} metric with 4 iterations in 3h 20min, though they also disappear in 5h 36min when increasing iterations for the ΔE^*_{76} metric from 4 to 8.

Fluorescent dyes. Using real-world measurements improves the realism of the fluorescence model, although some approximations are still required in order to limit memory and computation resources. Figure 5.22 shows that with our eight dyes from figure 5.16 we achieve a similar gamut as the analytical model from section 5.3.3. The gamut could be of a different shape or size for a different set of fluorescent dyes.

Including this fluorescence model in the coefficient cube optimizer shows promising results, though some artifacts remain and computation time is higher due to the brute force optimization over dye pairs. Figure 5.22 compares our result to the analytical fluorescence model.

Alternative interpolation. Figure 5.18 shows that interpolating the feature based coefficients (shift, width and slope) produces more perceptually meaningful results than interpolating the original coefficients (c_0, c_1, c_2). Our approach works well with bright, saturated colors, as long as all colors have the same sign of the slope value, but produces artifacts for flipping slope signs.

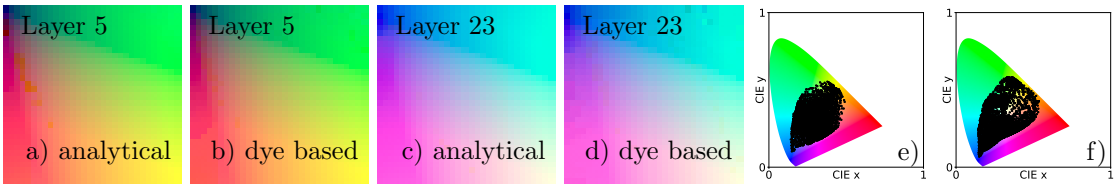


Figure 5.22: Comparison of the analytical (a, c, e) and dye based (b, d, f) fluorescence model without interpolation. The cube grid layers for $B = 0.161$ (a, b) and $B = 0.742$ (c, d) reveal artifacts in the dye based model. Figures e and f show the chromaticity of all colors represented in a 32^3 coefficient cube with a maximum slope of 0.01, only including colors with brightness of $X + Y + Z \geq 1.5$.

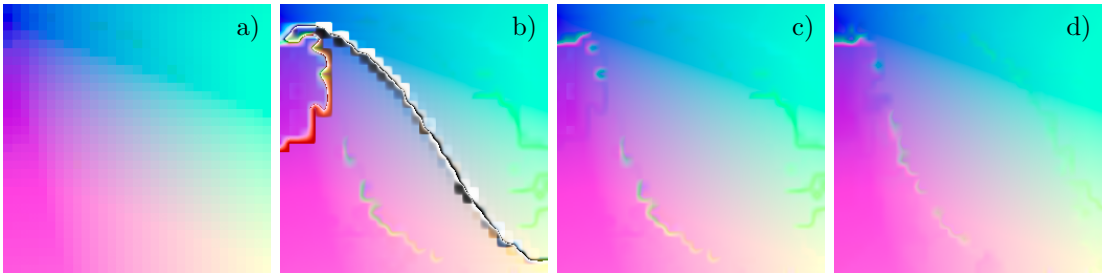


Figure 5.23: Cube layer for $B = 0.7$ and feature based coefficients with no (a), feature based (b), hybrid for different slope signs (c) and original coefficient (c_0, c_1, c_2) (d) interpolation. Interpolating slopes with different signs leads to distortions.

Figure 5.23 visualizes a slice of an optimized coefficient cube. As discussed in section 5.5.3, interpolating our coefficients leads to heavy artifacts, which can be avoided by the presented hybrid interpolation approach. Note that the discontinuities in the left image without interpolation are at the same position as the distortions in the images with interpolation.

5.6 Conclusion and Future Work

We presented a first method capable of automatically adding a fluorescent component to reflectance spectra that are synthesized based on RGB values. A typical use case for this are wide gamut input textures, such as textures in the ACEScg space which we use for our examples. In such a large RGB space, not all color values have corresponding pure reflectance spectra, as parts of it lie outside the gamut of solid colors. We solve this problem by adding fluorescent components to reflectance spectra, which allows us to generate highly saturated solid colors without violating any laws of physics.

Our simple parametric model for the absorption and emission spectrum of the fluorescent component is suitable for texturing as it is based on three floating point coefficients. It matches the sigmoid spectra used for normal reflectance in our system insofar as it is also physically plausible, without directly corresponding to any particular set of real fluorescent dyes. Instead, it is designed to phenomenologically match the mirrored appearance of real absorption and emission spectra.

Interpolation of our model's coefficients, while producing reasonable results in most cases, produces unintuitive results on the border where the spectral reconstruction switches from plain to fluorescence enhanced. Adding lower amounts of fluorescence to less saturated colors, even if it would not yet be necessary, might result in a smoother drop of standard reflectance intensity, and yield more reasonable interpolation results. Additionally, enforcing adjacent cube entries to have similar emission peaks, even at a slightly increased error, might be necessary to improve interpolation results further.

The alternative parameterization for the sigmoid function space from section 5.5.3 allows for more meaningful interpolation of the sigmoid coefficients, but displays remaining artifacts in the optimized coefficient cube and requires more work to arrive at a stable solution.

Using measurements of a set of preselected fluorescent dyes instead of the analytical model produced minor artifacts in the optimization process, but provides a more realistic model and can help improve our ability to match the appearance of highly saturated real-world materials. For instance, to generate spectral data that plausibly mimics the appearance of printed materials, the reradiation matrices of only very few real fluorescent inks can be used. For more extensive matching with real-world dyes one could analyze more fluorescent dyes and their absorption and emission spectra with a PCA-based method.

In this initial version, our model does not produce overlapping absorption and emission spectra. This way, we dodge the question whether fluorescence will ever result in a shift to shorter wavelengths or not. Dealing with this question is orthogonal to the purpose of this work, so we deemed it best to skirt the problem entirely. After this project was finished we investigated this question further and discuss our findings in section 4.7.2. With reradiation in both directions, evaluation of indirect lighting effects might become more interesting if a certain amount of re-absorption of fluorescent emission takes place.

Acknowledgments

We acknowledge funding by the Czech Science Foundation under grant number 16-08111S.

6 Spectral Mollification for Bidirectional Fluorescence

This chapter is based on
A. Jung, J. Hanika, and C. Dachsbacher (2020) "Spectral Mollification for Bidirectional Fluorescence", In: *Computer Graphics Forum* 39.2

Fluorescence creates bright and saturated colors both in natural and artificial materials by shifting energy between light of different wavelengths. So far, rendering fluorescence for continuous wavelengths or combined with wavelength dependent path configurations has only been feasible using spectral unidirectional methods. This chapter introduces a regularization-based approach for supporting fluorescence in a spectral bidirectional path tracer. Our algorithm samples camera and light sub-paths with independent wavelengths, and when connecting them mollifies the BSDF at one of the connecting vertices such that it reradiates light across a whole range of wavelengths. We discuss arising issues such as color bias in early iterations, consistency of the method and MIS weights in the presence of spectral mollification. We demonstrate our method in scenes combining fluorescence and transport phenomena that are difficult to render with unidirectional or spectrally discrete methods.

6.1 Introduction

Due to the constraint of energy conservation, there is a limit on how bright a surface of a certain color saturation can appear if it only reflects light. With fluorescence, we can achieve more saturated colors beyond this limit, as we did in chapter 5. Including fluorescence into a rendering system requires special care, since the inelastic scattering process changes the wavelength of the traced photon. This can heavily affect the geometry of subsequent scattering events. For instance, dielectric materials have an index of refraction that typically varies with wavelength, thus changing the angle of refraction with wavelength. Even more dramatic changes are possible when a participating medium has chromatic extinction, i.e. the mean free path length varies with wavelength. It is thus relatively straightforward to include fluorescence in a unidirectional path tracer [103], but bidirectional methods pose some challenges. We provide an intuition on these challenges in the next section, and formalize the problem in section 6.2.1.

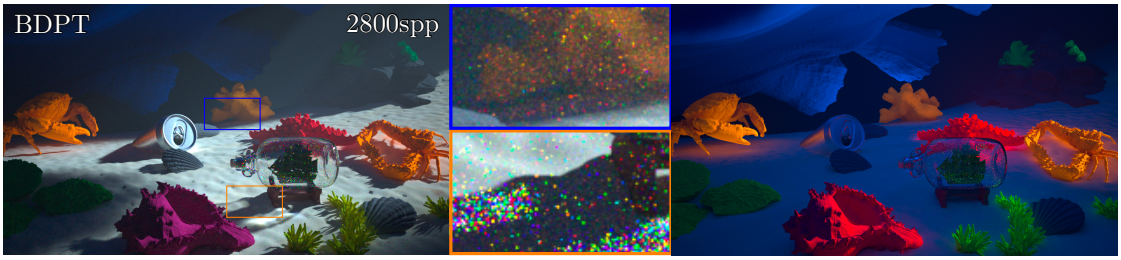


Figure 6.1: An underwater scene combining fluorescent surfaces and challenging illumination. It features smooth dielectrics (the bottle) and a rough conductor (the can) both with wavelength dependent index of refraction, and a homogeneous chromatic medium. The light source is a large but focused spot light, similar to caustic illumination under water, which is hard to render with unidirectional methods. The insets show a path tracer with next event estimation at equal time. **Right**: The scene is illuminated exclusively by UV light, **left**: visible and UV light. Both pictures are rendered with our spectrally mollified bidirectional path tracer, enabling bidirectional rendering in the presence of fluorescence.

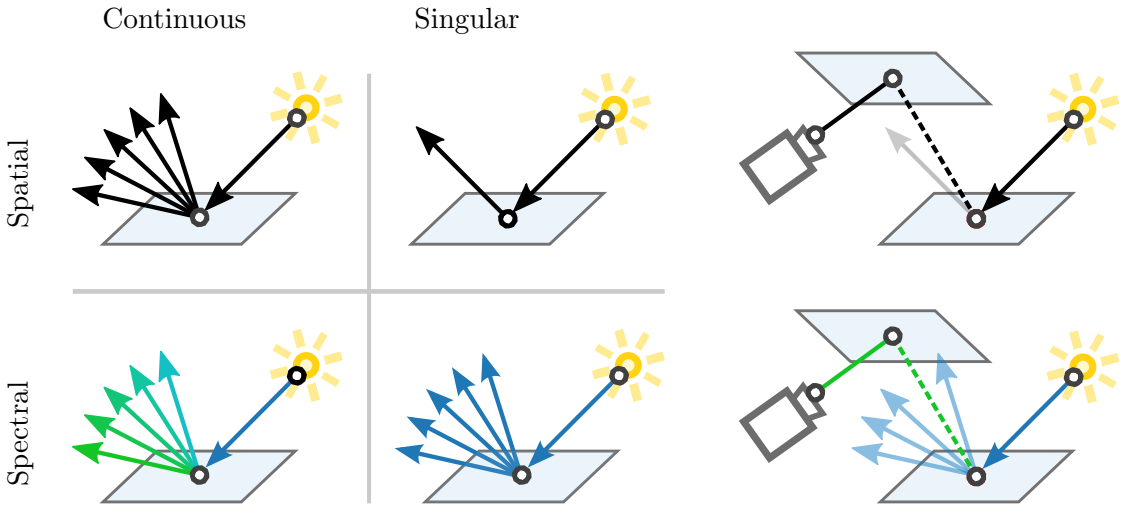
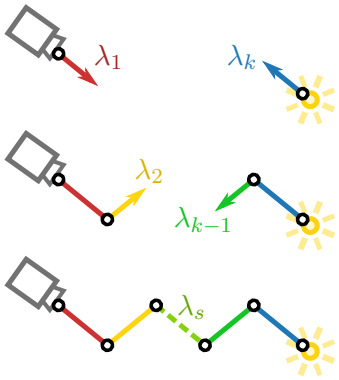


Figure 6.2: Just like specular surfaces (e.g. mirrors or glass) are singularities in the directional domain, non-fluorescent surfaces are singularities in the spectral domain. Bidirectional connections to surfaces with a specular component will always evaluate to 0 for the specular part of the surface's scattering behavior (**top right**), and if the camera and light sub-path start with different wavelengths they will also evaluate to 0 for the non-fluorescent part of any surface (**bottom right**).

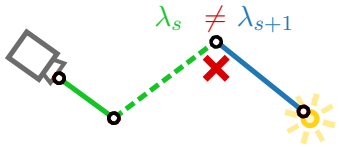
The problem with bidirectional fluorescence

First, let us consider a fully fluorescent path of the form $(\mathbf{x}_0, \lambda_1, \mathbf{x}_1, \lambda_2, \ldots, \lambda_k, \mathbf{x}_k)$ with fluorescent interactions at every surface. In order to sample this path with bidirectional path tracing, we need to start a camera and light sub-path with different randomly sampled wavelengths λ_1 and λ_k. Then, while we trace both sub-paths through the scene, we change the wavelength at every vertex, sampling a new wavelength from the absorption spectrum on the camera sub-path and from the emission spectrum on the light sub-path. Eventually, to connect the two sub-paths $(\mathbf{x}_0, \lambda_1, \ldots, \mathbf{x}_{s-1})$ and $(\mathbf{x}_s, \lambda_{s+1}, \ldots, \mathbf{x}_k)$, we somehow sample a wavelength λ_s for the connecting segment.



Then we evaluate the fluorescent BBRRDFs[1] at $\mathbf{x}_{s-1}$ and $\mathbf{x}_s$ for the wavelength shifts $(\lambda_s \to \lambda_{s-1})$ and $(\lambda_{s+1} \to \lambda_s)$. Assuming purely fluorescent surfaces at $\mathbf{x}_{s-1}$ and $\mathbf{x}_s$, these BBRRDF evaluations are well defined, and reasonably likely to be non-zero (as long as the wavelengths are covered by the fluroescent spectra's support).

However, if either of the surfaces at $\mathbf{x}_{s-1}$ or $\mathbf{x}_s$ is not fluorescent, we have a problem: Since all wavelengths along the path are continuous random variables that were sampled independently, both wavelengths λ_{s-1} and λ_{s+1} on the final sub-path segments will always be different from the connection segment's wavelength, and a non-fluorescent surface does not scatter any light between different wavelengths.



As figure 6.2 shows, this means that any path that is the result of a bidirectional connection will always have contribution 0 for any non-fluorescent interaction at one of the connecting vertices. This is analogous to how any path sampled from a bidirectional connection will always have contribution 0 for any specular interaction at either connecting vertex, since the specular (e.g. mirror-reflected) direction at the sub-path end vertex can never align with the connecting segment's direction.

We could try to solve this for a particular connection by deterministically setting the connecting segment's wavelength to the last wavelength of the sub-path with the non-fluorescent end vertex as in figure 6.3, but this still fails if both sub-paths end in non-fluorescent surfaces. Or we could sample the sub-paths without wavelengths first, and only sample wavelengths in a unidirectional manner once the sub-paths are connected, but this results in bad importance sampling if the path geometry strongly depends on the wavelength (e.g. in some participating media), and fails completely at sampling paths with wavelength-dependent singular geometry (e.g. from wavelength-dependent specular refraction).

[1] Reminder: The BBRRDF is the bispectral bidirectional reflection and reradation distribution function (see section 4.7.3) and describes the spectral and directional scattering behavior of a fluorescent surface.

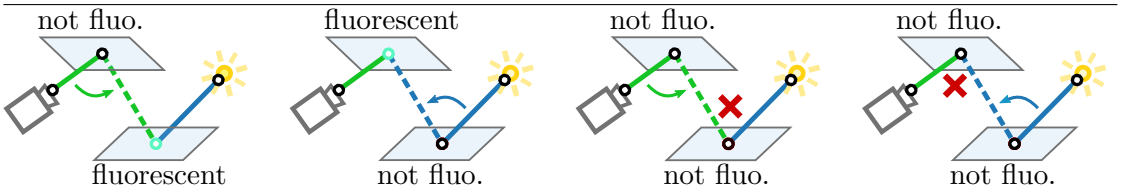
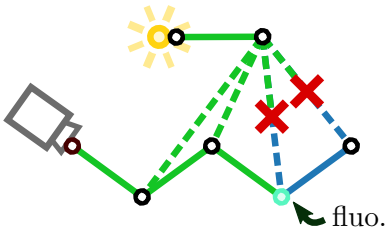


Figure 6.3: We can sample connections with one non-fluorescent vertex (**left**) by deterministically setting the connecting wavelength. If the end vertices of both sub-paths have a non-fluorescent component (**right**) this no longer works as one of them always evaluates to 0.

Finally, starting both sub-paths with the same wavelength as in the figure on the right does allow us to sample paths with exclusively non-fluorescent interactions. However, a single fluorescent interaction at either sub-path is enough to invalidate all connections from subsequent vertices to non-fluorescent vertices on the other sub-path.



Contribution

Figure 6.1 shows that in certain scene configurations bidirectional methods provide superior sampling quality. We propose a method to include fluorescence in a bidirectional path tracer by employing *mollification*. Somewhat counter-intuitively, in this setup the regular, non-fluorescent interactions pose a problem. We mollify these, i.e. we replace a singularity in wavelength domain by a finite-width normalized 1D function. Effectively, this adds a small amount of fluorescence-like behavior to every scattering event. The rendering algorithm is akin to a collision estimator, or to kernel estimation methods: if the wavelength on the sub-path from the eye is close enough to the wavelength of the sub-path from the light, the connection is performed. To result in a consistent estimator, i.e. to make the resulting bias disappear in the limit, we shrink the mollification width over time.

We discuss the computation of correct weights for multiple importance sampling (MIS), and the effect of mollification bias on non-fluorescent appearance.

In summary, the contributions of this chapter are:

- A bidirectional rendering method for fluorescence based on mollification.
- An analysis of mollification of reradiation matrices and continuous fluorescent spectra.
- A discussion of the corresponding MIS weights.
- An analysis of bias vs. variance tradeoffs.

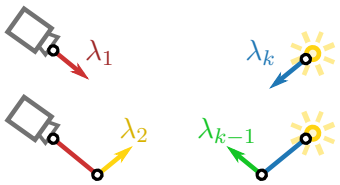
6.2 Background

Fluorescence. Section 4.7 describes the principles and modeling of fluorescence. In this chapter we use the fluorescent surface model from equation 4.7.1 [8], and for clarity we split the model into a fluorescent component f_{fluo} and non-fluorescent component f_r:

$$\begin{aligned} f_{fluo}(\omega_i, \lambda_i, \mathbf{x}, \lambda_o, \omega_o) &= \frac{1}{\pi} c \cdot a(\lambda_i) \cdot Q \cdot e(\lambda_o) \\ f_r(\omega_i, \mathbf{x}, \omega_o, \lambda) &= \frac{1}{\pi}(1 - c \cdot a(\lambda)) r(\lambda). \end{aligned} \tag{6.2.1}$$

We will only mollify the non-fluorescent component of the BBRRDF, and the methods we present also apply to mollifying the diagonal of a discrete 2D reradiation matrix. For better readability we will leave out the location parameter $\mathbf{x}$ where it is irrelevant.

Unidirectional fluorescence rendering. To render fluorescence with unidirectional path tracing or light tracing, we start with a vertex on the camera or light source, and then sample both a direction and a wavelength for each segment, potentially based on the current path prefix, as illustrated on the right.



Related work. Previous work on rendering fluorescence has been presented in section 4.7.5. To put this chapter in context, most previous work on fluorescence uses unidirectional methods [70, 103, 104, 99, 105, 71, 73]. In 2018 Jarabo et al. [100] proposed vectorized rendering to enable bidirectional transport with fluorescence. They evaluate path contributions for the full spectrum in discrete steps, whereas we use stochastic wavelength sampling for Monte Carlo integration of the wavelength domain. Their method ignores wavelength dependent refraction angles and segment lengths in participating media, which our approach supports natively without modifications.

6.2.1 Spectral Singularities

In the context of the bispectral rendering integral (equation 4.7.2) non-fluorescent scattering poses a singularity. For non-fluorescent surfaces the inner integral

$$\int_\Lambda L_i(\omega_i, \lambda_i) f(\omega_i, \lambda_i, \lambda_o, \omega_o) \cos\theta_i \, \mathrm{d}\lambda_i$$

collapses to a single function evaluation

$$L_i(\omega_i, \lambda_o) f_r(\omega_i, \omega_o, \lambda_o) \cos\theta_i$$

given the outgoing wavelength λ_o and the surface's (non-fluorescent) spectral BSDF f_r.

When rendering without fluorescence this singularity is avoided by omitting the integration over Λ. For non-fluorescent BDPT in particular, we first sample a single wavelength and then construct both the camera and light sub-path with this wavelength, allowing us to connect the (non-fluorescent) end vertices and guarantee that the wavelengths of the segments match up for a well-defined BSDF evaluation (see section 4.6 for more details on spectral rendering, and section 4.5.4 for more details on BDPT).

In a fluorescent context however, there exists no function that we can insert in the inner integral to achieve this behavior, as discussed in section 4.8. In other words, there is no BBRRDF $f_{nonfluo}$ such that

$$\int_\Lambda L_i(\omega_i, \lambda_i) f_{nonfluo}(\omega_i, \lambda_i, \lambda_o, \omega_o) \cos\theta_i \, \mathrm{d}\lambda_i = L_i(\omega_i, \lambda_o) f_r(\omega_i, \omega_o, \lambda_o) \cos\theta_i.$$

This is analogous to how a perfectly specular BSDF causes a geometrical singularity in non-spectral rendering, where there exists no BSDF $f_{specular}$ such that

$$\int_\Omega L_i(\omega_i) f_{specular}(\omega_i, \omega_o) \cos\theta_i \, \mathrm{d}\sigma(\omega_i) = L_i(\omega_{or}) a(\omega_o)$$

with a specular albedo $a(\omega_o)$ for the requested singular specular direction ω_o.

In order to be able to use the bispectral rendering equation for both fluorescent and non-fluorescent behavior we use the Dirac notation (see equation 4.8.3) to define

$$\int_\Lambda L_i(\omega_i, \lambda_i) \delta_{\lambda_o}(\lambda_i) f_r(\omega_i, \lambda_i, \lambda_i) \cos\theta_i \, \mathrm{d}\lambda_i := L_i(\omega_i, \lambda_o) f_r(\omega_i, \omega_o, \lambda_o) \cos\theta_i. \tag{6.2.2}$$

This means we can express the parametric BBRRDF from equation 6.2.1 as

$$f(\omega_i, \lambda_i, \lambda_o, \omega_o) = \frac{1}{\pi} \begin{cases} \delta_{\lambda_o}(\lambda_i)(1 - c \cdot a(\lambda_i)) r(\lambda_i) & \text{if } \lambda_i = \lambda_o \\ c \cdot a(\lambda_i) \cdot Q \cdot e(\lambda_o) & \text{else.} \end{cases}$$

Note that this expression is no longer a valid function and is only defined when inserted in an integral over Λ as above. Also, this formulation does not allow the fluorescent component to re-emit light at its original wavelength, but since this only affects the fluorescent part of the BBRRDF within a zero-set $\{\lambda_o\}$ of the spectral domain Λ, this does not change the result of the integration of f over Λ in the bispectral rendering integral (equation 4.7.2).

Spectral singularities with unidirectional path sampling. With unidirectional path construction algorithms (i.e. path tracing or light tracing) we handle these spectral singularities just like we handle directional singularities, as illustrated by figure 6.4. To start a path we sample a wavelength and starting direction, and trace a ray to find the next vertex in the scene. If that vertex is purely fluorescent, we sample a new wavelength and direction and proceed. If that vertex is not fluorescent, we copy the wavelength of the previous segment to the next segment, similar to how we would deterministically set the direction of the next segment for a specular surface. If the vertex has both a fluorescent and a non-fluorescent component, we first perform a discrete sampling decision to pick either a fluorescent or a non-fluorescent interaction with probabilities P_{fluo} and $P_{nonfluo}$, just how we would pick between a specular and non-specular interaction at a surface with multiple directional lobes. In case of a non-fluorescent direction we then copy over the wavelength and accumulate $P_{nonfluo}$ to the path's overall PDF, and in case of a fluorescent direction we sample a new wavelength with PDF $p(\lambda)$ and accumulate $P_{fluo} \cdot p(\lambda)$ to the path's PDF.

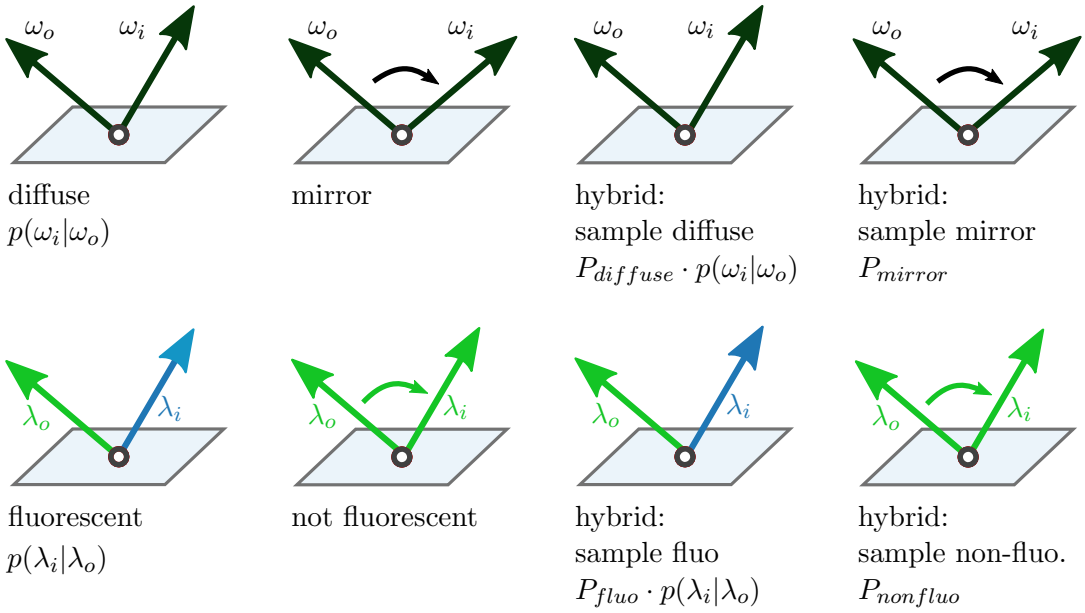


Figure 6.4: Local sampling decisions for non-singular, singular, and hybrid surfaces. For non-singular surfaces we sample a continuous PDF for a new value, i.e. a new direction (**top**) or wavelength (**bottom**) for the directional and spectral domain, respectively. With purely singular surfaces, the current value determines the next value. With hybrid surfaces, we first decide what kind of interaction to sample with a discrete probability, and then proceed accordingly.

Issues with singularities in bidirectional path sampling. With bidirectional path construction we need to start the camera and light sub-path with individual wavelengths and potentially change the wavelengths along the paths at fluorescent surfaces, just like we would change directions at every surface vertex that is not just transparent without refraction. When we connect a camera and light sub-path vertex, the two vertices determine the connection direction, but we still need to sample a wavelength for the connection. The two obvious choices are to sample a wavelength at either the camera or light sub-path vertex, just as we did during path construction (see figure 6.4).

Either way though, the paths' initial wavelengths as well as any fluorescent wavelength changes along the way were sampled from a continuous PDF. This means it is theoretically impossible that the wavelength sampled at the camera (or light) sub-path vertex is equal to the wavelength on the last segment of the light (or camera) sub-path, which also means that any non-fluorescent behavior at the light (or camera) vertex cannot be sampled this way (see figure 6.2 bottom). This is analogous to the case of a specular camera sub-path vertex, where the direction of the connecting segment does never align with the direction that would specularly scatter (e.g. be mirror-reflected or refracted) in the direction of the last camera sub-path segment (see figure 6.2 top).

We resolve this issue with mollification (see section 4.8) of the spectral singularity.

6.3 Spectral Mollification for Bidirectional Fluorescence

We now present our method for mollifying BBRRDFs to connect camera and light sub-paths for bidirectional path tracing. We start by discussing different strategies for mollifying the spectral singularity in full reradiation matrices (section 6.3.1) and in the non-fluorescent component of the separable BBRRDF from equation 6.2.2 (section 6.3.2). Depending on the mollifier, this may cause energy loss toward invisible wavelengths, or violate energy conservation by creating excess energy. Then, even though the mollified BBRRDFs are only evaluated but never sampled, we also need to be able to compute mollified probability density functions (*PDFs*, section 6.3.3) corresponding to the mollified BBRRDFs in order to combine the different sampling strategies inside a BDPT (section 6.3.4) by computing multiple importance sampling (MIS) weights (section 6.3.5). The mollifier which replaces the singularity further needs to shrink over iterations in order to keep the estimator consistent, which we outline in section 6.3.6.

6.3.1 Mollifying Reradiation Matrices

When working with reradiation matrices, we can choose to mollify the full matrix by using a 2D blur kernel on the 2D matrix, or we mollify the diagonal only. The second option is preferable, since expanding and blurring the full reradiation matrix is expensive and leads to undesired issues with energy conservation. We discuss this in more detail in appendix B.1. For the purpose of performing bidirectional connections, it is sufficient to mollify the non-fluorescent part of the BBRRDF, i.e. the diagonal of the reradiation matrix. In our implementation, we chose to work with an analytic BBRRDF model [8] that allows us to isolate and mollify only the non-fluorescent part.

6.3.2 Mollifying a Non-Fluorescent BBRRDF

In order to use a non-fluorescent BSDF in the bispectral rendering equation we express it as

$$f(\lambda_i \to \lambda_o, \omega_i \to \omega_o) = \delta_{\lambda_o}(\lambda_i) r(\lambda_i = \lambda_o, \omega_i, \omega_o) \tag{6.3.1}$$

consisting of a symbolic Dirac δ-component and some non-fluorescent reflectance which may depend on incident and exitant direction $\omega_{i,o}$, but only on one wavelength.

Applying a mollifier turns this BSDF in $[1/sr]$ into a BBRRDF in $[1/(sr \cdot nm)]$. We can base our mollification kernels on the indicator function

$$\mathbb{1}_X = \begin{cases} 1 & \text{if } X \\ 0 & \text{otherwise.} \end{cases}$$

So for example, a simple mollifier with radius d to replace $\delta_{\lambda_o}(\lambda_i)$ could be $\mathbb{1}_{|\lambda_o - \lambda_i| < d}/2d$, yielding

$$f_{moll}(\lambda_i \to \lambda_o, \omega_i \to \omega_o) = \frac{\mathbb{1}_{|\lambda_i - \lambda_o| < d}}{2d} r(\lambda_?, \omega_i, \omega_o).$$

However, this leaves the question how the spectral reflectance $r(.)$ should be evaluated. We consider three options. The first two are to evaluate $r(.)$ for either λ_i or λ_o, resulting in a piecewise constant mollifier for either a given λ_o or λ_i. The third one is a box filter based mollification to create a reciprocal BBRRDF. Examples for all three are shown in figure 6.6.

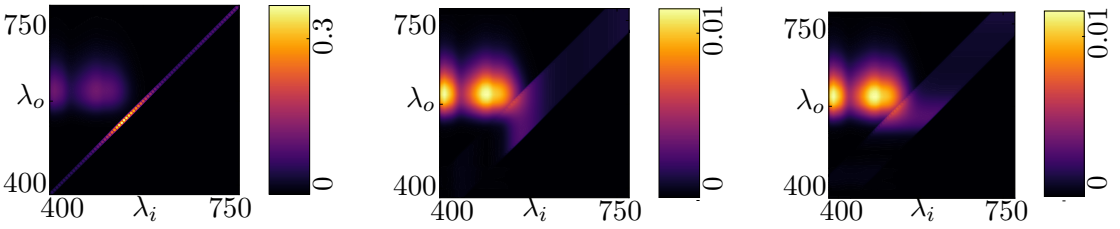


Figure 6.5: **Left**: An example reradiation matrix, wavelength values are in $[nm]$. **Center**: $r(\lambda)$ mollified with f_{mo}. **Right**: $(1 - ca(\lambda))r(\lambda)$ mollified with f_{mi} $(d = 50)$.

Constant mollification. Evaluating $r(.)$ for λ_o corresponds to *gathering* energy reflected to λ_o by $r(\lambda_o)$ over multiple incident wavelengths $[\lambda_o - d, \lambda_o + d]$:

$$f_{mi}(\lambda_i \to \lambda_o) = \frac{1}{2d} r(\lambda_o) \mathbb{1}_{|\lambda_i - \lambda_o| < d}. \tag{6.3.2}$$

Figure 6.5 (right) shows how this affects the corresponding reradiation matrix. This strategy has the advantage that, under a constant light source, the apparent color of the surface remains unchanged, as this mollifier can be considered to blur incident light. However, if we apply this strategy to hybrid surfaces with an additional fluorescent component the result may not be energy conserving anymore (see appendix B.1).

The second option, which we decided to use, is

$$f_{mo}(\lambda_i \to \lambda_o) = \frac{1}{2d} r(\lambda_i) \mathbb{1}_{|\lambda_i - \lambda_o| < d}. \tag{6.3.3}$$

This evenly reradiates incident energy at λ_i over a range $[\lambda_i - d, \lambda_i + d]$ according to $r(\lambda_i)$ (figure 6.5 center). The main advantage of this strategy is that it is fairly simple to implement as well as energy conserving, even in conjunction with a fluorescent components, since

$$\int_\Lambda f_{mo}(\lambda_i \to \lambda_o) \, \mathrm{d}\lambda_o = r(\lambda_i) \int_{\lambda_i - d}^{\lambda_i + d} \frac{1}{2d} \, \mathrm{d}\lambda_o = r(\lambda_i).$$

In appendix B.2 we show that it is also photon conserving. If only a full reradiation matrix is available, we recommend mollifying the diagonal using f_{mo}, which corresponds to mollifying $(1 - ca(\lambda))r(\lambda)$ instead of just r.

2D box filter mollification. Both constant mollification strategies result in a non-reciprocal BBRRDF with discontinuities at the mollification borders. We can produce a continuous and reciprocal mollified BBRRDF by applying a two-dimensional box filter with side length $b = 2d$

$$g_b(\lambda_i, \lambda_o) = \frac{\mathbb{1}_{|\lambda_i - \lambda_o| < b/2}}{b^2}$$

to r, which yields

$$\begin{aligned} f_{mb} &= [f * g_b](\lambda_i \to \lambda_o, \omega_i \to \omega_o) \\ &= \frac{1}{b^2} \int_{max(\lambda_i, \lambda_o) - b/2}^{min(\lambda_i, \lambda_o) + b/2} r(\lambda, \omega_i, \omega_o) \, \mathrm{d}\lambda, \end{aligned} \tag{6.3.4}$$

where $\int_x^y \dots$ evaluates to 0 if $x > y$. While this can be implemented as a 2D box filter operation on the reradiation matrix diagonal, we still only mollify in one dimension for either a fixed λ_i or fixed λ_o.

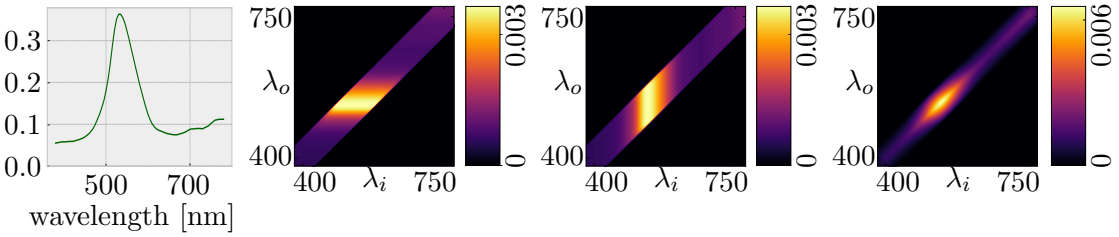


Figure 6.6: The reflectance spectrum (**left**) of a non-mollified non-fluorescent BSDF as in equation 6.3.1, which is then mollified to create a BBRRDF over incident (equation 6.3.2, **center left**) and exitant (equation 6.3.3, **center right**) wavelengths ($d = 50$), and the box filtered BBRRDF (equation 6.3.4, **right**).

Clamping. All three options leak energy out of both ends of the visible range. It is possible to construct clamped versions of f_{mo}, f_{mi} and even f_{mb}, to conserve all energy in the visible range. However we decided against it because a) the mollifier would depend on prior knowledge about the "visible wavelength range" at the camera and b) a clamped BBRRDF can no longer reflect ultraviolet light which might be converted to visible light by some fluorescent surface later on the path. Clamping to visible plus ultraviolet light instead would in turn require prior knowledge about the minimum wavelength emitted in the scene, and might effectively result in an unclamped BBRRDF for other light sources.

One-sided kernel. We also tested the kernel

$$f_{asym}(\lambda_i \to \lambda_o) = \frac{1}{2d} r(\lambda_i) \mathbb{1}_{\lambda_i < \lambda_o < \lambda_i + 2d} \tag{6.3.5}$$

with the intent of creating a BBRRDF more akin to a fluorescent BBRRDF, as real fluorescent materials typically convert shorter to longer wavelengths. However, figures 6.13, 6.15 and 6.17 reveal that this kernel causes much stronger color bias.

6.3.3 Mollified PDF

The PDF $\bar{\mathbf{p}}$ corresponding to a mollified BBRRDF f_{moll} is defined as any other PDF:

$$\bar{\mathbf{p}}^c(\lambda_i|\lambda_o) = \frac{f_{\text{moll}}(\lambda_i \to \lambda_o)}{\int_\Lambda f_{\text{moll}}(\lambda \to \lambda_o)\,\mathrm{d}\lambda}$$

for camera paths, and

$$\bar{\mathbf{p}}^l(\lambda_o|\lambda_i) = \frac{f_{\text{moll}}(\lambda_i \to \lambda_o)}{\int_\Lambda f_{\text{moll}}(\lambda_i \to \lambda)\,\mathrm{d}\lambda}$$

for light paths. Depending on how the BBRRDF was mollified, these equations result in different terms with different simplifications, which we discuss in detail in appendix B.3. In general, we denote spectrally mollified PDFs as $\bar{\mathbf{p}}(\lambda_x|\lambda_y)$. In case of constant mollification with f_{mo} we get

$$\bar{\mathbf{p}}^c_{mo}(\lambda_i|\lambda_o) = \frac{\mathbb{1}_{|\lambda_i - \lambda_o| < d}\, r(\lambda_o)}{\int_{\lambda_o - d}^{\lambda_o + d} r(\lambda)\,\mathrm{d}\lambda} \quad \text{and} \quad \bar{\mathbf{p}}^l_{mo}(\lambda_o|\lambda_i) = \frac{\mathbb{1}_{|\lambda_i - \lambda_o| < d}}{2d},$$

and analog PDFs for f_{mi}. This reveals an issue of our mollification technique, which is that for given ω_i, ω_o we need to be able to integrate the mollified BBRRDF over wavelengths in order to compute the correct PDF. However, since we only evaluate the mollified BBRRDF at a connection vertex, but never sample from it, we do not actually need to evaluate the correct PDF. Instead, we will only require a mollified PDF for computing MIS weights, which also works (though less optimally) with approximate values.

In case of the box mollification f_{mb} we get analog PDFs $\bar{\mathbf{p}}^c_{mb}(\lambda_i|\lambda_o)$ and $\bar{\mathbf{p}}^l_{mb}(\lambda_o|\lambda_i)$. In this case there are no more simplifications and we end up with a double integration over $r(.)$ in the denominator. The full equation and a solution for piecewise linear spectra $r(.)$ are presented in appendix B.3.

6.3.4 Sampling Strategies

Path sampling works analogously to regular BDPT. We sample initial wavelengths to start a path from the camera and a path from the light sources, and trace both paths. We associate each edge of a path with an individual wavelength. During the random walk, new wavelengths are sampled at fluorescent interactions. A camera path samples the absorption spectrum, a light path samples the emission spectrum. NEE samples an additional wavelength for the connecting segment from the last vertex on the path before the connection (see figure 6.7).

Inner connections (with more than one vertex on both camera and light sub-paths) sample the wavelength of the connecting segment from the camera sub-path. To make the connection work, the last light sub-path vertex is mollified spectrally. This decision is arbitrary, instead we could also mollify at the last camera vertex. The connection is rejected if the wavelengths are too far apart.

Thus the spectral PDF of a connected path with $k+1$ vertices and a connection from $\mathbf{x}_s$ to $\mathbf{x}_{s+1}$ is

$$p(\lambda_1)p(\lambda_2|\lambda_1)\cdots p(\lambda_{k-1}|\lambda_k)p(\lambda_k),$$

and the spectral PDF for an implicit (i.e. path traced) or NEE path is

$$p(\lambda_1)p(\lambda_2|\lambda_1)\cdots p(\lambda_k|\lambda_{k-1}).$$

6.3.5 Multiple Importance Sampling Weights

Kaplanyan et al. [20] only mollify paths that cannot be sampled by any bidirectional strategy. Their maximum distance heuristic minimizes overall bias by assigning weight 1 to the least biased mollification option and 0 to all other strategies.

In our test scenes any path can be sampled by pure forward path tracing (i.e. there are no singular emitters). Furthermore, especially if there is no fluorescence along the path, their strategy of selecting the path with the longest connection distance does not translate well to mollifying in the wavelength domain as opposed to the geometry domain.

We want to make full use of all bidirectional regularized sampling strategies and therefore use MIS weights similar to the balance heuristic (equation 4.3.5):

$$w_{s,t}(\mathbf{X}) = \frac{\bar{p}_{s,t}(\mathbf{X})}{\sum_{i=0}^{s+t} \bar{p}_{i,s+t-i}(\mathbf{X})}, \tag{6.3.6}$$

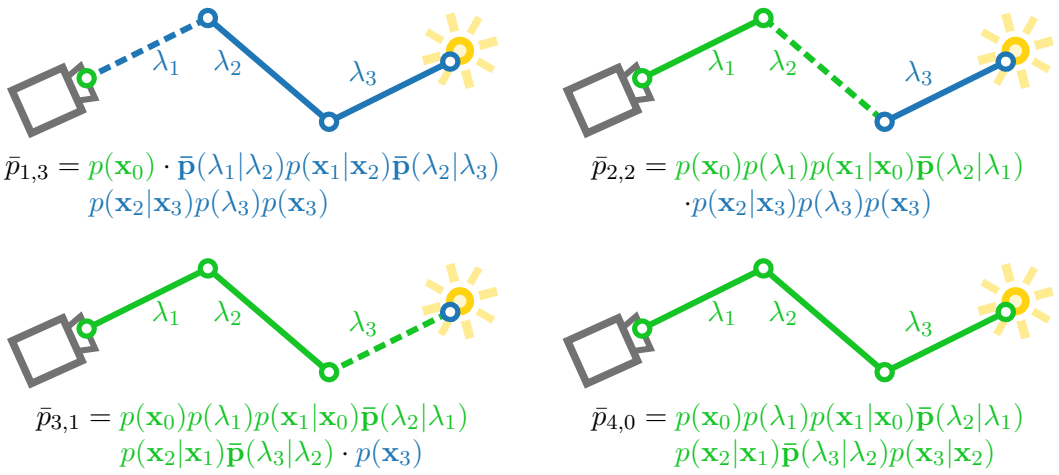


Figure 6.7: Mollified PDFs $\bar{p}_{s,t}$ used in MIS weight computations for camera (green) and light (blue) sub-paths. We index the camera vertex as $\mathbf{x}_0$ and the light vertex as $\mathbf{x}_{s+t-1}$.

where $\bar{p}_{s,t}(\mathbf{X})$ is the mollified PDF of a path $\mathbf{X}$ formed by connecting an s-vertex camera sub-path to a t-vertex light sub-path, with the mollified PDF $\bar{\mathbf{p}}$ instead of the actual sampling PDF for all but the first wavelength of each sub-path (see figure 6.7). This means we pretend that any wavelength except the first along each sub-path was sampled from the mollified BBRRDF at the previous vertex, and evaluate the PDF corresponding to the mollified BBRRDF.

By construction, the weights for a given path sum up to one. This also ensures that each PDF $\bar{p}_{s,t}$ has the same measure in the sense that they all cover the same number of spectral PDFs, even if there is no fluorescent event along the path. Alternatively, path space would have to be extended to consider fluorescent and non-fluorescent events at the same vertex as an additional dimension. This makes the analysis a lot more complex for layered surfaces which can be both at the same time. On the other hand, in our approach most of the mollification factors at the vertices will cancel out in the weight computation in equation 6.3.6.

Since we use the wavelength of the camera sub-path for the connecting segment, we evaluate the mollified PDF at $\mathbf{x}_{s-1}$ for λ_s when computing $\bar{p}_{s,t}$ (see figure 6.7).

6.3.6 Consistency of the Estimator

Kaplanyan et al. [20] show how to obtain a consistent estimator by decreasing the support of the mollifier over iterations depending on dimensionality. Since we only mollify one dimension once per path, we need to decrease the sequence of mollification distances d_n as $\mathcal{O}(n^{-1}) < d_n < \mathcal{O}(1)$. As suggested in their work we decrease d_n as

$$d_n = d_0 n^{-s},$$

which satisfies above condition for $s \in (0, 1)$. In section 6.4.6 we evaluate different choices of d_0 and s, including $s = 0$ (i.e. $d_n \equiv d_0$).

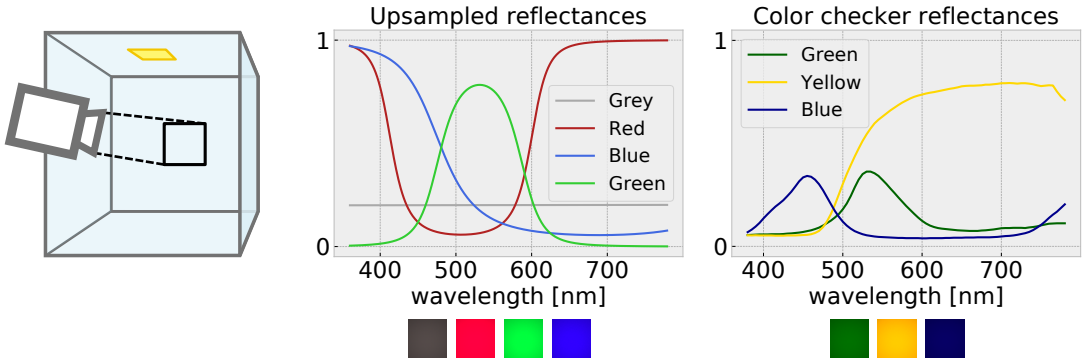
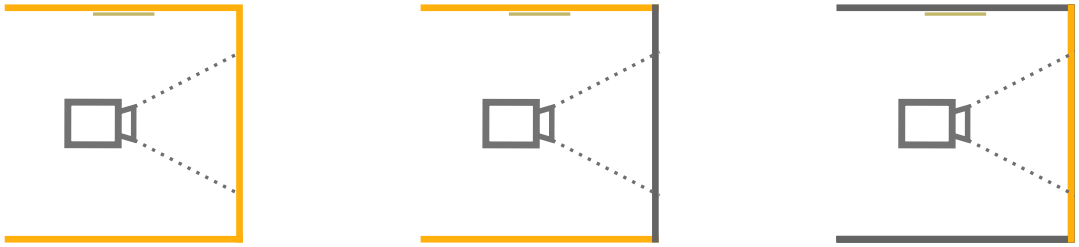


Figure 6.8: Geometry and reflectance spectra of the color bias tests. The first four spectra were upsampled from RGB=[0.2,0.2,0.2] (gray), [0.8,0.05,0.15] (red), [0.05,0.8,0.15] (green) and [0.05,0.15,0.8] (blue) using the method by [6]. The last three are color checker measurements from [128].



a) b) c)

Figure 6.9: Setups for the color bias tests, with a) all walls with the same reflectance (figure 6.13), b) color reflectances on all walls except a gray back wall (figure 6.15), and c) colorful reflectance only on the back wall, all others being constant gray (figure 6.14).

6.4 Evaluation

Unless mentioned otherwise we use constant mollification as in f_{mo} (equation 6.3.3), an initial mollification distance of $d_0 = 100$nm and a shrinking coefficient of $s = 1/4$. We approximate the mollified non-fluorescent PDF $\bar{\mathbf{p}}^c_{mo}(\lambda_i|\lambda_o) \approx \frac{\mathbb{1}_{|\lambda_i-\lambda_o|<d_n}}{2d_n}$ for MIS weights.

6.4.1 Scene Setups

Spectra
We show the reflectance and fluorescence spectra for the scenes in figures 6.8, 6.10 and 6.11. As we use the spectra-based BBRRDF model from equation 4.7.1, we show the spectra directly instead of the corresponding reradiation matrices.

Empty Box
We evaluate color bias with an empty box, constant light source and different combinations of colored and gray walls (figure 6.9) without fluorescence. Figure 6.8 shows the reflectance spectra and geometry. The camera only sees the center of the back wall.

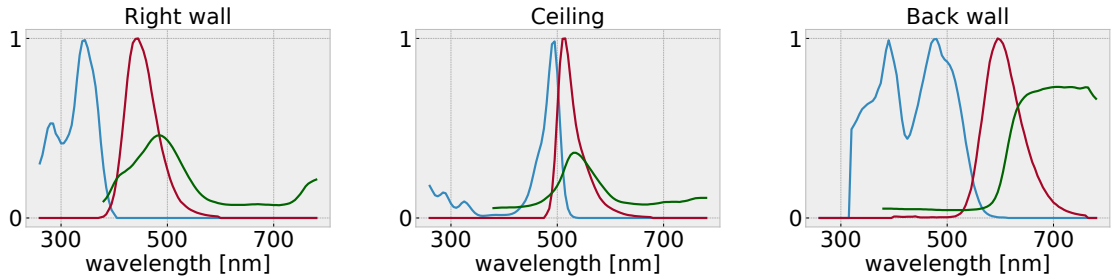


Figure 6.10: Reflectance (green) and fluorescence absorption (blue) and emission (red) spectra in the spheres scene. The fluorescence measurements are from [69], the spectra are color checker measurements for cyan, green and red from [128].

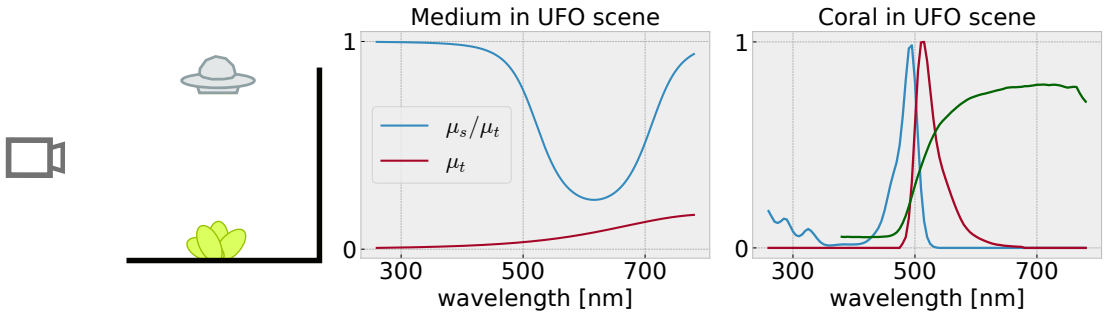


Figure 6.11: UFO scene geometry and spectra. **Left**: Volume scattering albedo (red) and μ_t (blue). **Right**: Reflectance (green) [128] and fluorescence absorption (blue) and emission (red) spectrum [69] of the coral.

Spheres Scene

Figures 6.20, 6.21 and 6.22 show scenes with dielectric and conductor spheres. The ceiling and left wall are diffuse and non-fluorescent with the gray reflectance from figure 6.8, the remaining planes are fluorescent with spectra described in figure 6.10.

UFO Scene

Figure 6.11 (left) depicts the geometry of the UFO scene. The scene is filled with a homogeneous medium with wavelength-dependent mean free path and scattering coefficients depicted in figure 6.11 (center). The spectra of the fluorescent coral in the center are shown in figure 6.11 (right). The top of the UFO is a smooth dielectric with wavelength-dependent index of refraction (IOR), with Abbe number (dispersive power) 10 and $n_D = 1.3$ [129]. The UFO body is a smooth conductor based on the complex IOR of silver. The left and right wall are the same material as the right wall in figure 6.10. The floor, back wall and ceiling are constant gray as in figure 6.8.

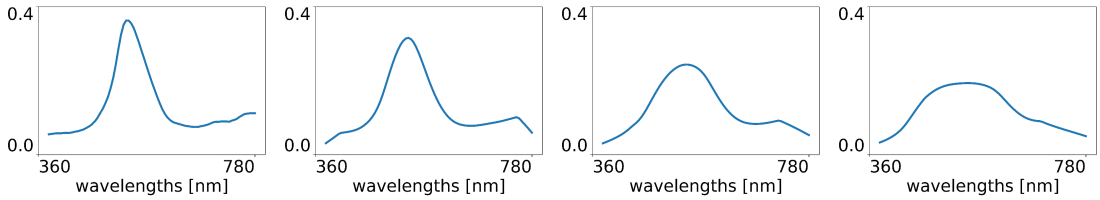
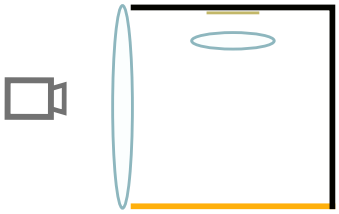


Figure 6.12: Original reflectance (**left**) and apparent reflectance after mollification with a constant kernel with increasing radius. The mollification blurs the spectrum, which usually causes a desaturated appearance, and causes energy loss toward invisible wavelengths, which can darken the appearance.

Lens Scene

The figure on the right depicts the geometry of the lens scene. The scene contains no fluorescence to allow comparisons to BDPT without mollification. The lenses are smooth dielectrics with Abbe number 23 and $n_D = 1.3$ The floor is upsampled [6] from $RGB = [0.9, 0.7, 0.1]$. The light source emits a constant spectrum with a highly focused emission profile, manifesting in the bright spot on the floor. The remaining walls are diffuse gray as in figure 6.8.



In this setup NEE is completely useless, as the specular lenses prohibit connections to camera or light vertices. Here, we can only use sub-paths with at least 2 vertices each.

6.4.2 Color Bias

Scene setup. We evaluate color bias in an empty Cornell box with gray or colored walls, as depicted in figure 6.9. In order to isolate the bias as much as possible we only show the back wall rendered with (2,2)-paths, i.e. paths with length 4 that were created by connecting a 2-vertex camera sub-path to a 2-vertex light sub-path. We chose this setup since these paths are the only ones where we can guarantee that mollification always happens, and it always happens at either a side wall, the floor, or the ceiling. If unidirectional sampling techniques contribute to the image as well, the overall bias of the image is expected to be (much) lower.

Causes of color bias. Our method produces color bias for two reasons. First, light bleeds to nearby wavelengths, thus blurring the involved spectra as shown by figure 6.12, which desaturates the appearance (see figures 6.13 and 6.15). Yet figure 6.16 reveals that even with constant reflectances and illumination, where blurring is not an issue, some bias remains.

The reason for that is that camera paths start with an initial wavelength λ_c from some predefined visible range $[\lambda_{min}, \lambda_{max}]$, and light paths start with λ_l from $[\lambda_f, \lambda_{max}]$, where λ_f is the smallest emitted (ultraviolet) wavelength. Assuming λ_l and λ_c are sampled uniformly, we plot the probability of $|\lambda_l - \lambda_c| < d$ in figure 6.18. It shows that the probability for sampling a λ_l within range d decreases for λ_c close to the borders of the visible range (or toward longer wavelengths only, if UV light is supported). As a result, mollified images are biased toward colors in the center of the visible spectrum and lack

red (and violet/blue, if no UV light is supported, as in figure 6.18 right). With a small mollification d, this effect is hardly noticeable, since colors at the boundary of what we use as visible spectrum ($[360, 780]$ nm) are hardly visible anyway, but for larger d (or if the visible range is smaller to begin with), this color bias becomes increasingly noticeable (see figure 6.16, and figure 6.18 center). Conceptually, this type of bias resembles boundary bias in photon mapping.

Bias for (non-)constant reflectances. In figure 6.13 all surfaces have the same colored reflectance. Mollification happens at the floor, ceiling, or either of the side walls, although the apparent bias is somewhat weakened by the second colored reflection off the back wall.

In figure 6.14 only the back wall has a colored reflectance, and the remaining surfaces have a constant gray reflectance spectrum. In this setup mollification for $(2, 2)$-paths only happens for the constant spectrum. This has a negligible effect on the appearance.

In figure 6.15 all walls except the back wall have the same colored reflectance, and the back wall has a constant gray reflectance spectrum. In this setup mollification for $(2, 2)$-paths always happens on the colored surface, and the resulting bias is preserved by the constant reflectance spectrum on the gray back wall as seen by the camera, instead of being weakened by a colored back wall as was the case in figure 6.13.

Connection acceptance rate. Figure 6.18 plots the probability that a 2-vertex camera path for λ_c and a 2-vertex light path for λ_l in a scene without fluorescence are successfully connected using mollification as in equation 6.3.2, provided a connection is geometrically possible. This also shows that the lower λ_f, the lower the probability for accepting the resulting path. This does not cause bias since the resulting path contribution includes a division by $p(\lambda_f)$, which decreases accordingly, but it does cause higher noise levels.

Boundary bias. We could avoid boundary bias using a kernel which explicitly depends on λ_o. For example, this can be achieved by clamping the mollified reradiation matrix. We discuss this option and why we decided against it at the end of section 6.3.2.

Consistency. As explained in section 6.3.6 we shrink d over time. Therefore any color bias will disappear in the limit. With a centered kernel, even at $d = 50$nm, which is still quite large, the bias is hardly visible.

One-sided kernel. We test the one-sided kernel from equation 6.3.5 in figures 6.13, 6.14 and 6.15 on the right-hand side, and in figure 6.17. Instead of blurring light over both shorter and longer wavelengths as the other kernels, which mostly affects saturation and brightness, this kernel only shifts light in one direction, thus deviating strongly from the reference hue. This effect is strongest in figure 6.15, where light mollified on a colored surface is reflected off a gray surface (instead of off a same-colored surface as in figure 6.13, which weakens the hue distortion).

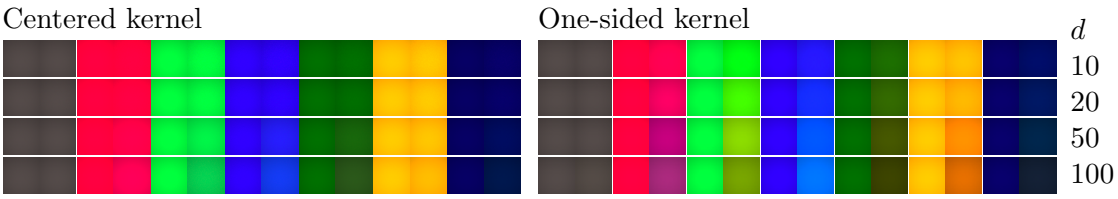


Figure 6.13: These renders show the back wall of a uniformly colored Cornell box, rendered with a PT with NEE for 4-vertex-paths (left columns) and rendered exclusively with mollified $(2,2)$-paths (right columns). The mollified paths use a centered (equation 6.3.3, **left block**) or one-sided (equation 6.3.5, **right block**) constant mollification kernel (i.e. $s = 0$) with $d \equiv 10, 20, 50, 100$nm (**top to bottom**). Figure 6.8 shows the reflectance spectra in this scene. With constant spectra (i.e. the gray on the left), only boundary bias is an issue, which is hardly noticeable even at large d. The bleeding bias can become strong for colorful spectra, but considering the visible wavelength range covers roughly 400nm, radii like 50nm or 100nm are impractically large anyway. While the centered kernel mostly results in darkened colors for extreme radii, the one-sided kernel noticeably affects the hue.

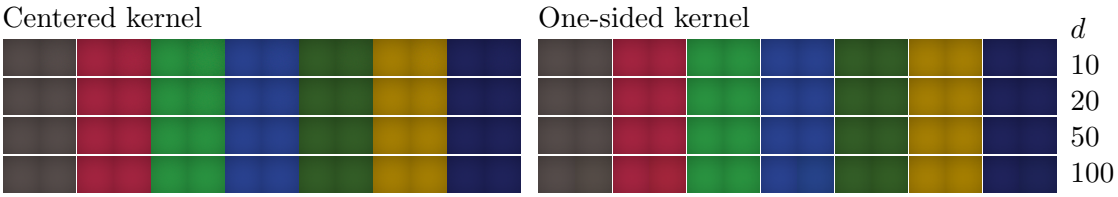


Figure 6.14: Same setup as figure 6.13, but all walls except the back wall have a constant gray reflectance. This means mollification always happens on surfaces with a constant spectrum, where boundary bias is the only issue.

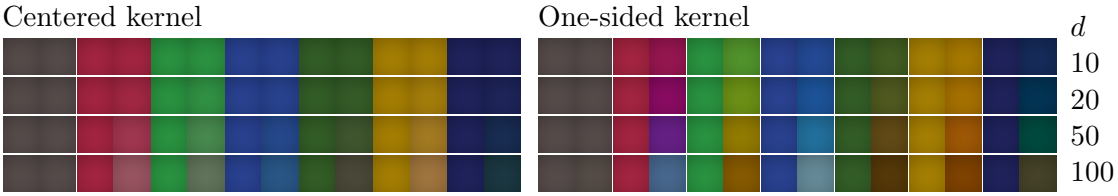


Figure 6.15: Same setup as figure 6.13, but the back wall has a constant gray reflectance. In this setup mollification always happens at one of the colored walls, where it blurs the spectrum and causes boundary bias, and we can indirectly see the mollified appearance reflected by the gray wall.

6.4.3 Constant Mollification vs. Box Filtering

We could not detect any significant difference between box filtered mollification as in equation 6.3.4 and constant mollification as in equation 6.3.3. Figure 6.19 shows one such example with fluorescent walls and ultraviolet illumination rendered with both methods.

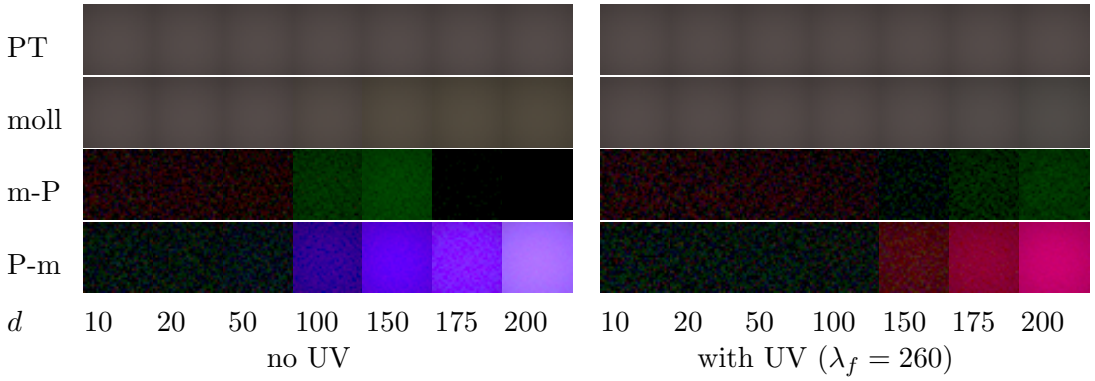


Figure 6.16: Boundary bias, illustrated by rendering a gray Cornell box with $(2, 2)$-paths (**second row**), no shrinking (i.e. $s = 0$), compared to a path traced reference (**top**). The bottom rows show signed difference images multiplied by 64 (**third row**: Mollified - PT, **fourth row**: PT - mollified). On the right, the constant illuminant includes ultraviolet light ($\lambda_f = 260$nm), so the boundary bias (initially) only affects long wavelengths.

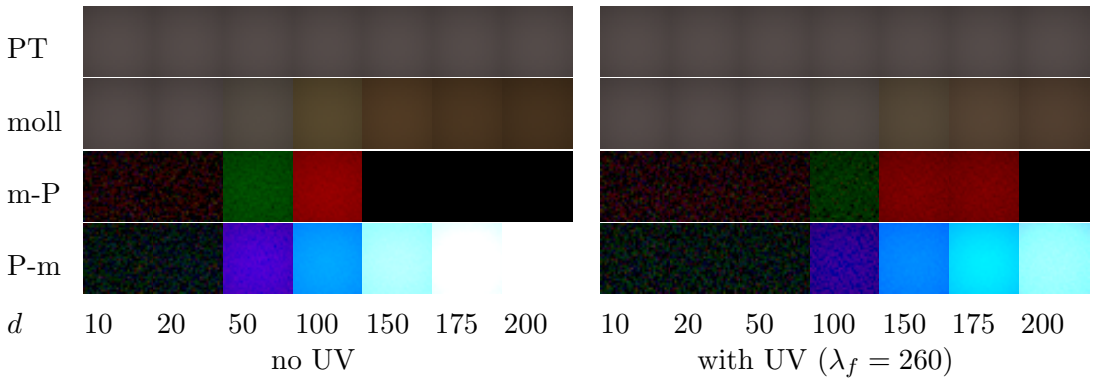


Figure 6.17: Same setup as figure 6.16 but for the asymmetric kernel ($\mathbb{1}_{\lambda_i<\lambda_o<\lambda_i+2d}/2d$).

6.4.4 Bidirectional vs. Unidirectional

Depending on the scene different sampling strategies (PT, NEE, LT, or BDPT) work best, We present three scenes in which our method is superior to a regular PT with NEE (figure 6.1, figure 6.20, figure 6.23). If simple PT/NEE is better than BDPT (e.g. if all paths are easily sampled unidirectionally) in a scene without fluorescence, PT/NEE will also be better than our mollified BDPT in that scene with added fluorescence.

Fluorescence and smooth dielectrics. Figure 6.20 shows a scene with both fluorescent and non-fluorescent gray walls containing smooth dielectric and metal spheres with wavelength dependent indices of refraction. The light source has a constant emission spectrum which extends to the ultraviolet range. Our BDPT does not connect to the perfectly specular spheres, so mollification only occurs for paths with two consecutive bounces off the walls, floor or ceiling. Since this scene is dominated by paths that are easily sampled by next event estimation from camera and light paths, mollification plays only a small role in this setup. Nevertheless our algorithm allows us to render the scene with a full BDPT.

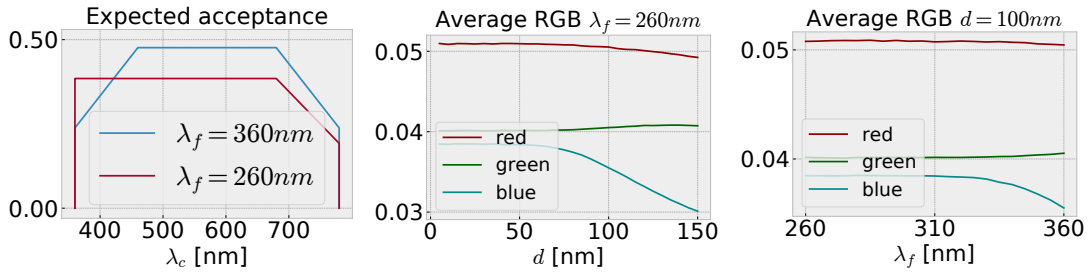


Figure 6.18: **Left**: Probability of uniformly sampling a wavelength λ_l for a light path from $[\lambda_f, 780]$ such that the distance to the wavelength from the camera $|\lambda_l - \lambda_c| < d =$ 100nm, for two choices of λ_f: with and without UV light. **Center, right**: Average image RGB for the gray wall scene in figure 6.16 for varying mollification radius d (center) and minimum sampling wavelength λ_f (right) .

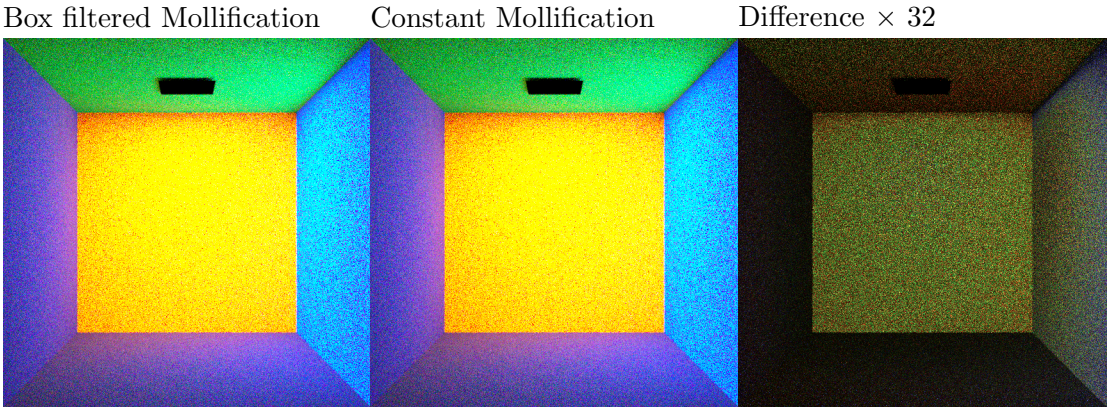


Figure 6.19: **Left**: Mollification of the non-fluorescent reflectance $r(\lambda)$ using a box filter as described in equation 6.3.4. **Center**: Constant mollification of $r(\lambda)$ as in equation 6.3.3. **Right**: Difference times 32.

Fluorescence and mollified metals. Figure 6.21 shows a scene containing rough metals that are spectrally mollified by f_{mo} as in equation 6.3.3. They are modeled as a microfacet BSDF using a GGX distribution, with a wavelength-dependent complex index of refraction. We do not provide an RMSE plot for this scene since even at over 100k spp, the PT+NEE reference implementation has not properly converged (orange inset in figure 6.21). Instead, equal sample and time comparisons show the superiority of BDPT in this setup. In figure 6.22 we compare the same insets for $d_0 = 100$nm and a shrinking coefficient $s = 1/4$ to $d_0 \equiv 5$nm and $s = 0$.

Fluorescent surfaces in volumes. Similar to figure 6.1, figure 6.23 illustrates a path tracer's issues with non-fluorescent volumes combined with a focused light source. The light tracer works well for volume paths but struggles with the UFO and light source.

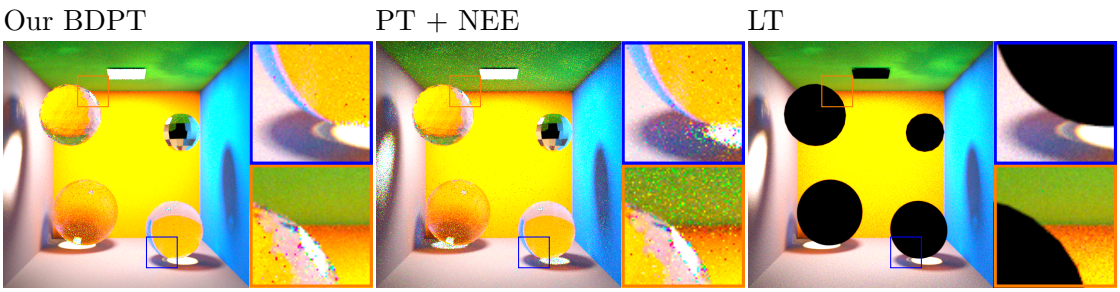


Figure 6.20: A scene with fluorescent and gray walls and smooth dielectric and metal spheres at 256 spp. **Left to right**: Our mollified BDPT, PT+NEE, LT.

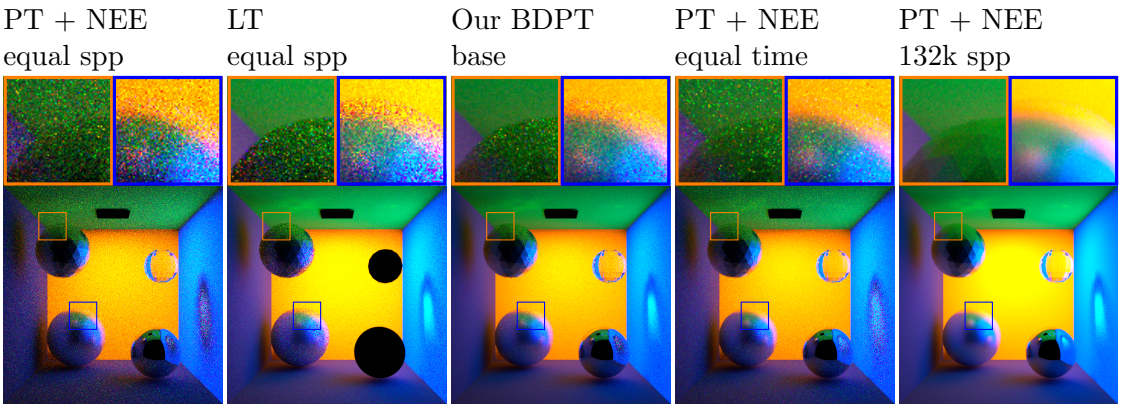


Figure 6.21: A scene with fluorescent and gray walls and rough conductors, illuminated by constant ultraviolet light. **Left to right**: PT + NEE (256 spp, 81 seconds), LT (256 spp, 85 seconds), our BDPT ($d_0 = 100$nm, $s = 1/4$, 256spp, 448 seconds), PT + NEE roughly equal time (2048 spp, 625 seconds), PT + NEE 132k spp. The caustic on the right wall is caused by UV light being refracted through the sphere, which is then made visible by the fluorescent blue material.

6.4.5 Mollified vs. Unmollified BDPT

In this section we compare our mollified BDPT to the unmollified base version. This is only possible in scenes without fluorescence. The contributions of strategies with at most one camera or light vertex are the same for both renderers. For all other strategies, the mollified BDPT rejects a certain number of connections based on the wavelengths of the sub-paths (also see figure 6.18 left), which leads to slower convergence. If we use constant mollification f_{mo} (equation 6.3.3) with current mollification distance d_n, the MIS weights of the mollified BDPT reduce to

$$\frac{\alpha_{s,t} p_{s,t}(\mathbf{X})}{\sum_{i=0}^{s+t} \alpha_{i,s+t-i} p_{i,s+t-i}(\mathbf{X})}, \tag{6.4.1}$$

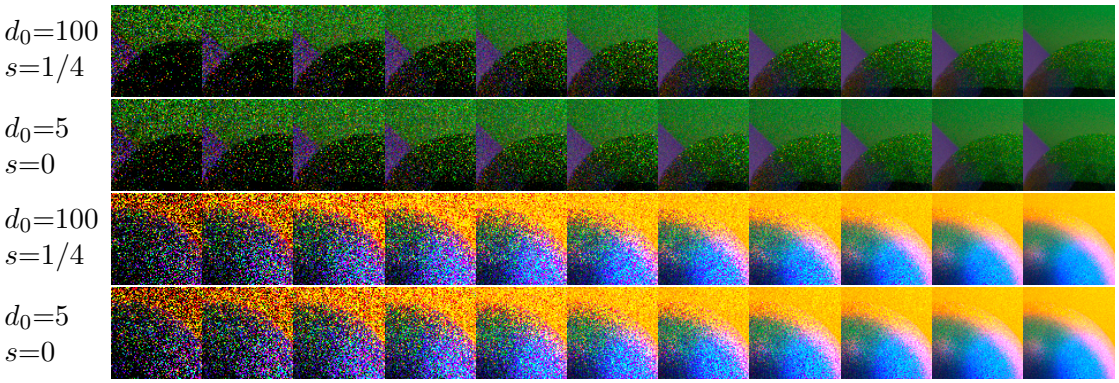


Figure 6.22: Orange and blue insets from figure 6.21 rendered with our BDPT at 1, 2, 4, ..., 1024 spp. **Top**: Starting with a lot of mollification and decreasing over time, $d_0 = 100$nm, $s = 1/4$. **Bottom**: Starting small without decreasing, $d_0 = 5$nm, $s = 0$. Except toward higher sample counts at the top two rows there is little noticeable difference.

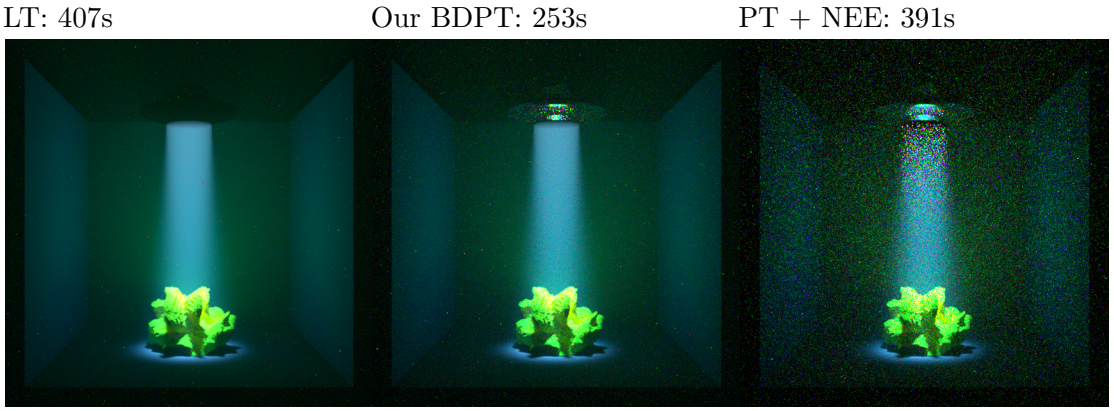


Figure 6.23: A specular UFO casting light on a sponge in a medium (setup details in figure 6.11). **Left**: LT, 1024 spp, 407 seconds. **Center**: Our BDPT, 64 spp, 253 sec. **Right**: PT + NEE, 1024 spp, 391 sec.

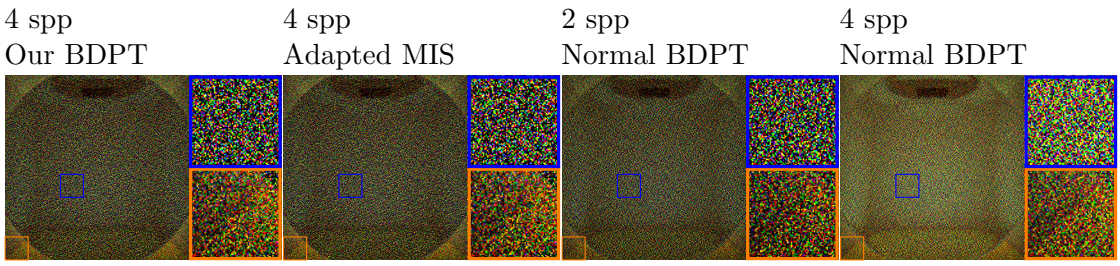


Figure 6.24: In the lens scene BDPT relies exclusively on connecting sub-paths with at least two vertices each. **Left to right**: Our BDPT (4spp, $d_0 = 100$nm), Our BDPT with adapted MIS weights (4spp, $d_0 = 100$nm), unmollified BDPT (2spp), unmollified BDPT (4spp). With $d_0 = 100$nm the regularized BDPT accepts close to 50% of all connections (see figure 6.18) and thus reaches similar noise with 4 spp as unmollified BDPT with 2 spp.

where $p_{s,t}(\mathbf{X})$ is the unmollified PDF used during sampling evaluated for the path $\mathbf{X}$, and

$$\alpha_{s,t} = \begin{cases} p(\lambda_1)p(\lambda_{s+t-1}) & \text{if } s,t > 1 \\ p(\lambda_1)/(2d) & \text{if } t \leq 1 \\ p(\lambda_{s+t-1})/(2d) & \text{if } s \leq 1. \end{cases}$$

Since typically $2d$ is (much) smaller than the range of wavelengths sampled at the camera (e.g. $[360, 780]$ nm) or light source (e.g. $[260, 780]$ nm), paths with at most one camera or light vertex get weighted overproportionally compared to the unmollified BDPT's MIS weights, and paths with at least 2 vertices on each sub-path receive lower weights.

We tried demonstrating this effect in the scene shown in figure 6.24, which relies heavily on such paths. However it is far outweighed by the slower convergence due to rejection of sub-paths with wavelengths that are more than d_n nm apart. The figure also contains an image with a hack replacing $\alpha_{j,k}$ with $\alpha_{1,k}$ wherever $i, j > 1$ to bring the mollified MIS weights closer to the unmollified weights. This shows close to no difference to the original mollified MIS weights in equation 6.4.1.

6.4.6 Convergence Rate

Scenes without fluorescence. Any scene without fluorescence that can be handled by our BDPT can also be rendered more efficiently by a typical unmollified BDPT. The unmollified version does not reject connections based on wavelengths but instead samples both sub-paths with the same wavelength.

Scenes with fluorescence. Variance reduction of our mollified BDPT can be improved by increasing the initial mollification distance d_0 so that fewer paths are rejected, and decreasing the shrinking coefficient s. Vice versa, bias in the long run is reduced by decreasing d_0 and increasing s. In figure 6.29 we show how d_n shrinks for different values of s. Section 6.4.2 shows bias depending on the current mollification distance d_n. Too aggressive shrinking, while theoretically consistent, results in too much variance to converge in reasonable time (figure 6.25 $s = 3/4$).

Root mean squared error (RMSE). Increasing the kernel width increases the number of successfully mollified paths, thus decreasing variance and initially decreasing the RMSE, as shown in figure 6.26. In particular, using an n times wider kernel results in roughly n times as many contributing paths. Therefore the mollified method with an n-times wider kernel achieves a similar RMSE with $1/n$-th as many samples (dotted lines), at least until variance decreases enough for bias to become apparent.

Parameter choice. Assuming finite time for rendering an image, it may be preferable to choose a fixed mollification distance for all iterations. In figure 6.22 we compare shrinking with $s = 1/4$ starting from $d_0 = 100$nm to keeping $d = 5$nm fixed from 1 spp to 1024 spp. In the shrinking progression $d_{1024} = d_0 \cdot 1024^{-1/4} \approx 18$nm does not quite reach 5nm, yet there is little noticeable difference, except maybe the upper left sphere. This is explained by the fact that this scene is dominated by paths that are easily sampled by NEE or the LT, and paths requiring mollification play only a minor role here.

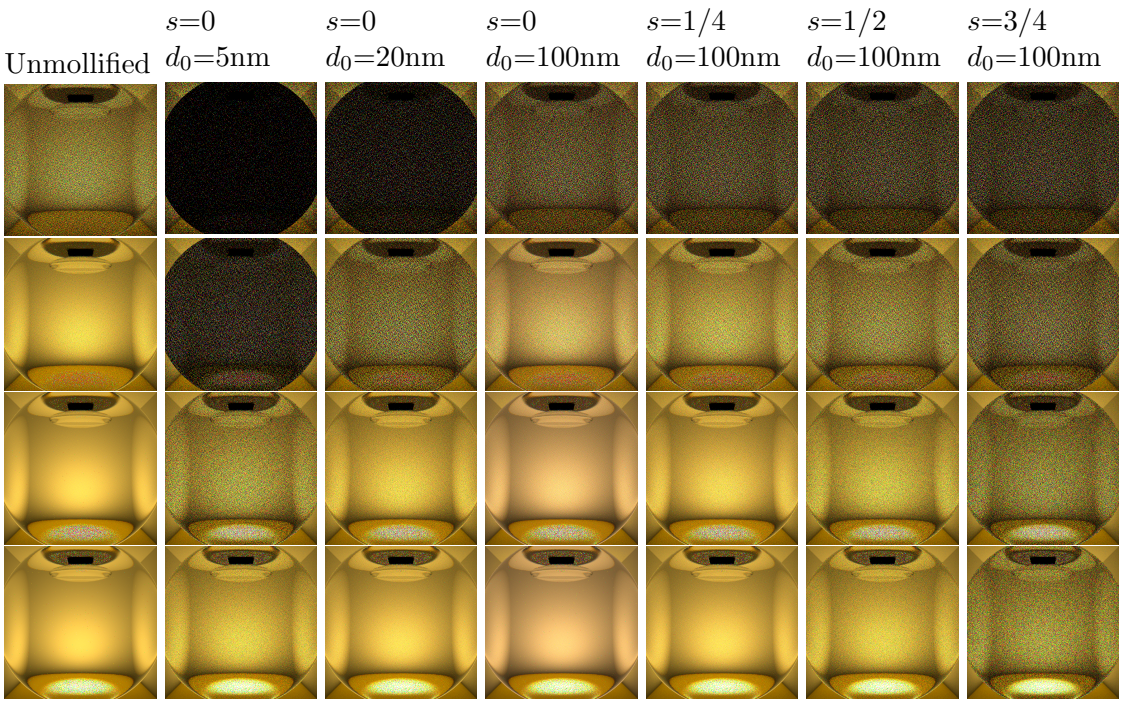


Figure 6.25: The lens scene rendered with 4, 32, 256 and 1024 samples per pixel (**top to bottom**). Left is the unmollified BDPT, next to it is the mollified BDPT with different values for d_0 and s. This illustrates the tradeoff between initial noise reduction due to higher d_0 and small s (**center, center right**) and long-term bias reduction with larger s and smaller d_0 (**center left**). Setting $s = 0$ is not actually consistent, but with a "good" choice of d_0 may yield better results in limited time.

In figure 6.25 mollified paths are more significant, and we test various combinations of d_0 and s. This clearly illustrates the trade-off between initial noise reduction and long term bias reduction. At only 4 samples per pixel, $d_0 = 100$nm without any shrinking proves to be a viable strategy for generating quick previews. At 32 spp, $d_0 = 100$nm, $s = 1/4$ outperforms $d_0 = 5$nm, $s = 0$. Then, at 1024spp, both combinations perform similarly well. On the other hand, $d_0 = 100$nm, $s = 3/4$, while initially better than $d_0 = 5$nm, $s = 0$, shows the most variance in the long run due to the drastic shrinking ($100 \cdot 1024^{-3/4} = 0.005$nm). In general, the optimal d_0 and s depend on the number of available samples.

Figures 6.27 and 6.28 show quantitatively that the optimal parameters depend on the scene settings, reflectance spectra and sample count.

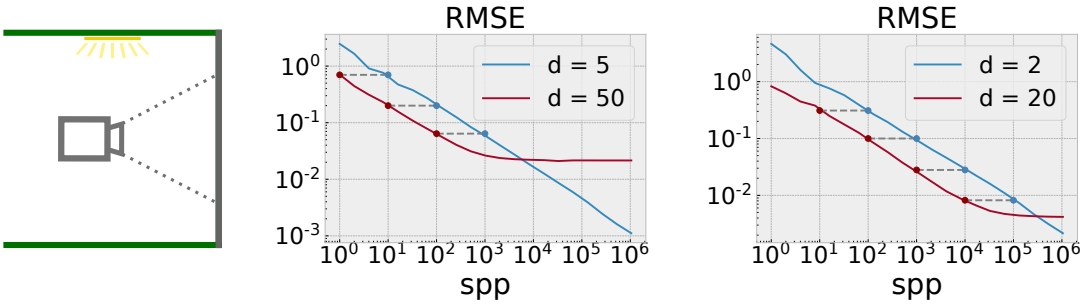


Figure 6.26: RMSE for an empty box scene (see figure 6.9b) with color checker green reflectance (figure 6.8 right), rendered with mollified $(2, 2)$-paths only and no shrinking ($s = 0$). With a 10-times smaller mollification radius d (blue), the variance of mollified techniques increases by a factor of roughly 10. Therefore a similar RMSE is achieved with 10 times as many samples (dotted lines), at least at lower sample counts before bias becomes significant and the lower radius outperforms the larger one (red).

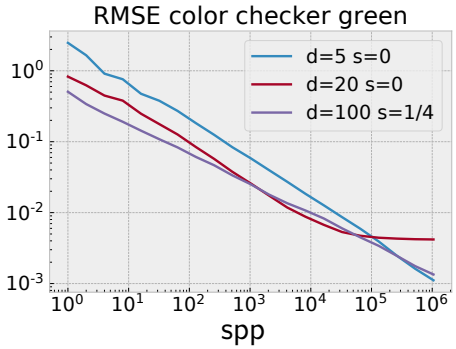


Figure 6.27: RMSE with same setup as figure 6.26, for different d_0 and s. For $d_0 = 100$ we chose $s = 1/4$ which performs best in figure 6.28 (right). Here, each combination outperforms the others in a certain wavelength range. At even higher sample counts, the $d \equiv 5$ technique will stop converging as it is biased, and the $d_0 = 100, s = 1/4$ combination will eventually have the lowest RMSE once again.

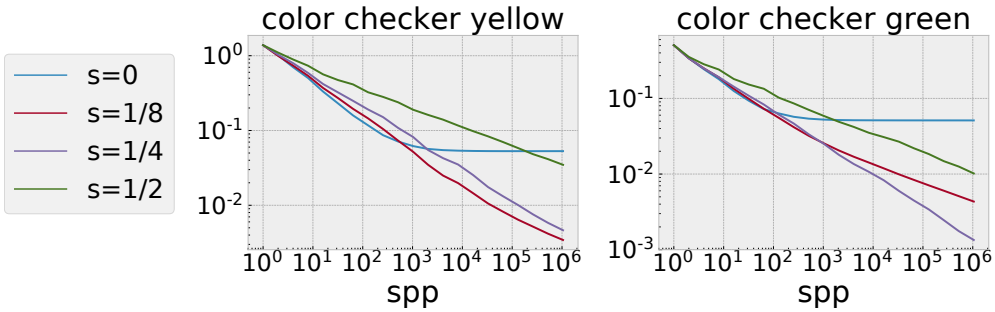


Figure 6.28: RMSE with same setup as figure 6.26, $d_0 = 100$nm and different shrinking parameters s. Initially, no shrinking is best since with larger d more paths are successfully mollified which decreases variance. In the long run, even in this simple setup, the best s depends on the samples per pixel and reflectance spectra involved.

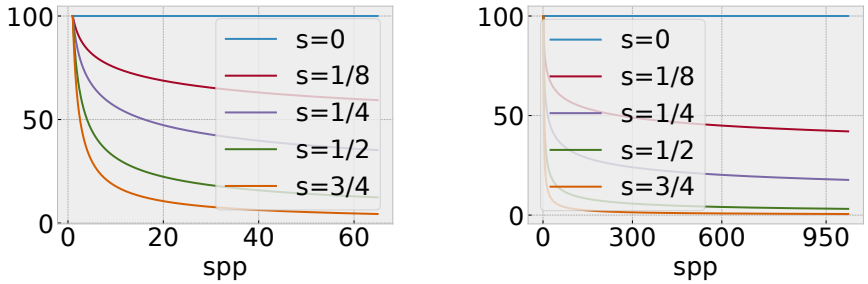


Figure 6.29: Mollification radii d_n for different shrinking parameters s over samples per pixel, starting from $d_0 = 100$nm.

6.5 Discussion

Alternatives. During the course of this work, we considered a few alternatives to mollification, in the hope of finding an unbiased solution. Brute force tracing a complete set of wavelengths (quantize the visible spectrum in 5nm steps, as the CIE color matching functions, for instance) does not help, since a wavelength dependent scattering event will break the path configuration for all but one wavelength. This results in a lot of wasted computation. There is also the other extreme: first sampling the path *completely without* a wavelength and later re-evaluating it for the required wavelengths. This strategy has the same issue with geometric configurations: a good path cannot be sampled without a wavelength.

Combination with hero wavelength sampling. The paths in our implementation transport single wavelengths, but combination with techniques that carry more than one wavelength at a fixed stratified offset [55] are straightforward: if mollification between the two main wavelengths is admissible, this also holds for the copies.

6.6 Limitations and Future Work

The biggest limitation of our work is the deteriorating performance of BDPT in the regular case without fluorescence, since many connections get discarded due to mismatching wavelengths. This is presumably a large proportion of the scenes currently encountered in practice. The problem is somewhat alleviated by the fact that light tracing and path tracing with next event estimation are not affected by this, only the inner connections require mollification. This could potentially be addressed by a specialized variant of the hero wavelength sampling scheme [55], or by applying our regularization scheme to a spectral probabilistic BDPT [130]. Another potential shortcoming is the use of approximate MIS weights, but we could not prove a big impact here.

Evaluating the mollified PDF requires integrating the mollified BBRRDF over a range of wavelengths. For most BBRRDFs, a separable approximation such as the one we used should work well. This may be interesting for rough dielectric transmission, where the direction and the wavelength show strong correlation. Interestingly, mollifying a non-fluorescent BBRRDF and designing new fluorescent BBRRDF models require similar mathematics.

The convergence of kernel estimation methods deteriorates with increasing dimensionality [20]. As we only mollify one dimension, our approach is in this regard more efficient than e.g. photon mapping which requires at least a 3D lookup. However, photon mapping methods leverage multiple cached sample points.

Another interesting direction is to derive optimal initial mollification distance and shrinking rate parameters. Previous work [36] shows that for photon mapping optimal parameters depend on the scene. Our experiments suggest that the situation is similar for the spectral domain. A straightforward extension is to combine our spectral mollification scheme with directional mollification [20] and spatial mollification [21].

6.7 Conclusion

We proposed a mollification-based method to combine bidirectional path sampling methods with wavelength changing, i.e. fluorescent materials. The mollification resolves the fundamental contradiction between sampling new wavelengths and connecting path vertices, but increases variance for inner connections at non-fluorescing surfaces as compared to regular BDPT. Thus, our method should only be used if fluorescent surfaces are present and if there is a clear gain from such sampling strategies (see figure 6.1). Our method relies on MIS, so it will transparently prefer PT or LT in simple cases. As with regular BDPT, additional inner path connections come at an overhead, but with the separable constant mollification we proposed there is very little additional raw performance overhead of our method over unmollified BDPT.

6.8 Acknowledgments

This work was funded by DFG grant DA 1200/6-1, project 323377784. The crab, seastar and shell models in figure 6.1 are downloaded from http://threedscans.com. The sponge and coral models are part of the digital atlas of ancient life by the paleontological research institution, Ithaca, New York https://sketchfab.com/DigitalAtlasOfAncientLife. The can model was created by `jackho` at http://www.blendswap.com/blends/view/74793.

7 Spatial Mollification for Next Event Estimation with Small Lights

This chapter is based on unpublished work.

Path tracing handles many light transport effects well, but struggles with close to singular integrands that are hard to sample, and completely fails whenever the integrand contains singularities that cannot be sampled at all. Regularization of the integrand, e.g. by increasing the roughness of specular surfaces or using piecewise constant BSDFs, replaces one or multiple (near-)singularities with smoother functions to enable sampling, or to further trade variance for bias.

In this chapter we propose a new regularization technique for next event estimation from specular surfaces based on the photon merge as in Vertex Connection and Merging [21]. In particular, we intersect the specular direction with a virtual plane around the light source, place a kernel around this intersection and accept an NEE sample on the light source if it is within that kernel. We use ray differentials to adapt our regularization kernel to the path prefix, and can guarantee a consistent method by shrinking the kernel over iterations. The kernel applies to point lights that would otherwise be impossible to sample, and it enables us to reduce the variance from sampling area lights in exchange for bias. The benefit of this trade-off depends on the relative size of the kernel and light source. We thus combine our regularized technique with BSDF sampling through MIS.

7.1 Introduction

Despite the general popularity of Monte Carlo path tracing techniques, there remain configurations that are problematic for Monte Carlo integration and even guided versions. Singularities and near-singularities pose one such problem, since the numerical solutions we use are completely unable to sample singularities, and struggle at reliably sampling near-singular settings. In this chapter we consider (near-)singularities in the reflectance of surfaces (e.g. perfect mirrors or glass), and in the light sources (e.g. directional emitters or point lights).

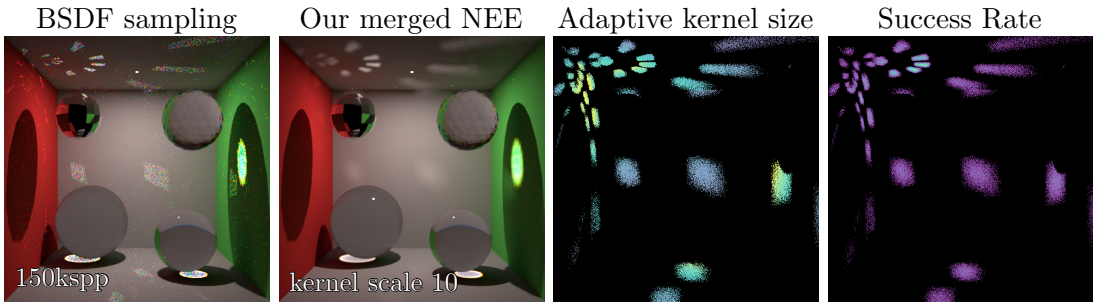


Figure 7.1: Path tracing with vanilla next event estimation (NEE) does not allow NEE connections from specular surfaces to the light source (**left**). Our regularized technique (**center left**) enables such connections if the NEE vertex is within an adaptive kernel (**center right**) around the specular direction, and combines their contributions with BSDF sampled paths through MIS. The percentage of successful connections (**right**) depends on kernel size and scene configuration.

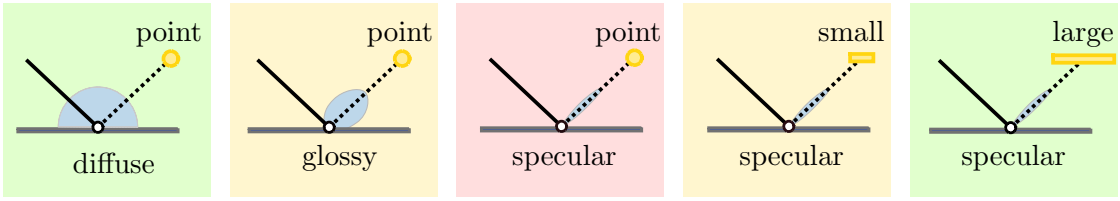


Figure 7.2: Green: easily sampleable by either NEE (**left**) or BSDF sampling (**right**). Yellow: Hard to sample. Red: Impossible.

In path tracing with next event estimation the camera can connect to a light source in two ways: by explicitly connecting to a light vertex (next event estimation, NEE), or implicitly hitting a light source by chance through sampling the surface reflectance described by the bidirectional scattering distribution function (BSDF). As both techniques importance sample only a part of the rendering integrand, they both suffer if the remainder of the integrand becomes increasingly influential. In particular, NEE becomes more noisy with increasingly peaky BSDFs, while BSDF sampling becomes more noisy with smaller light sources (figure 7.2). Once the BSDF or light source becomes truly singular, the corresponding technique fails completely, and paths hitting a specular surface before a singular emitter cannot be sampled at all.

The last case (figure 7.2 red) can be solved by other sampling strategies such as light tracing, or by regularization [20, 131], replacing either one or both singularities with a smooth function that can then be integrated. In configurations that are difficult but theoretically possible to sample (figure 7.2 yellow), regularization can trade noise for bias, either by regularizing a technique that already works, or by enabling additional techniques that would otherwise be impossible.

Some previous online regularization approaches merge vertices of camera and light sub-paths in the scene. Vertex Connection and Merging even showed how to combine such merge paths with paths that were constructed through unregularized BSDF sampling with multiple importance sampling (MIS). In this chapter, we show how to extend these concepts to NEE by shifting the merge location from scene vertices to light source vertices. This enables us to perform NEE with both specular and near-specular surfaces.

The variance of our technique depends on the size of the regularization kernel, i.e. the spatial extent of the vertex merges, whereas the variance of BSDF sampling depends on the light source size. We thus use multiple importance sampling (MIS) to combine our technique with paths where BSDF sampling hits the light source by chance, to minimize the variance of the overall estimate by assigning higher weights to whatever technique exhibits lower variance.

7.2 Background

In section 4.5 we used Monte Carlo integration to solve the rendering integral (section 4.4.6) for scattered radiance L_s

$$L_s(\mathbf{x}, \omega_o) = \int_{\mathbb{S}^2} L_i(\mathbf{x}, \omega_i) f_s(\omega_i, \mathbf{x}, \omega_o) \cos\theta_i \, \mathrm{d}\sigma(\omega_i). \tag{7.2.1}$$

However, we can only apply Monte Carlo integration if there is an actual integral to solve. In case of singular surfaces (mirrors, glass, ...) or light sources (point lights, directional lights) there are no functions f_s or L_i for which the rendering integral evaluates to the actual outgoing radiance, as discussed in section 4.8. This means that we cannot use the rendering integral to model such configurations, and have to either extend what we mean by "integral" to allow for singularities in the integrand, or look for other ways to describe this behavior.

In section 4.8.1 we discuss these options in more detail, but most importantly, we can use the symbolic Dirac notation (equation 4.8.3), which complicates scenarios with multiple singularities (e.g. a point light viewed through a mirror, see appendix C.1.2), or we can express singularities in the context of distributions (section 4.8.2), either using the Dirac distribution to describe point lights and singular surfaces directly, or taking a non-singular distribution family to the limit (equation 4.8.4) to describe how a regularized singularity can converge to a true singularity.

7.2.1 Regularization

Mollification can regularize configurations that are hard or impossible to sample, as described in section 4.8.3.

Kernel density estimation approximates a function at some location $\mathbf{x}$ by integrating the function weighted with some kernel $\mathcal{K}$ over the surroundings $\mathbb{K} := supp(\mathcal{K})$ of $\mathbf{x}$:

$$f(\mathbf{x}) \approx \int_{\mathbb{K}} \mathcal{K}(\mathbf{x}, \mathbf{y}) f(\mathbf{y}) \, \mathrm{d}\mathbf{y}.$$

The resulting integral can then be numerically computed with Monte Carlo integration. For example, photon mapping [33] uses kernel estimation to approximate the outgoing radiance by accumulating radiance of incident photons.

Discussion. Having singularities in the integrand means that some of the integral's dimensions collapse and we cannot integrate over all dimensions. Regularizing such singularities via mollification reintroduces these dimensions back to the integral, and

they collapse back to singularities when the mollification diminishes. Kernel estimation on the other hand can be interpreted as adding more dimensions to a problem, where a previously continuous function is now a collapsed singularity which is then resolved with mollification.

For example, we can view an ordinary BSDF $f_s(\omega_i, \mathbf{x}, \omega_o)$ as a singular subsurface scattering BSDF $f_{ss}(\omega_i, \mathbf{x}_i, \mathbf{x}_o, \omega_o)$, since SSBSDFs distribute light continuously around the area of the entry point $\mathbf{x}_i$, whereas normal BSDFs only scatter light at a single surface position $\mathbf{x}_o = \mathbf{x}_i$. Replacing a normal BSDF f_s with a SSBSDF

$$f_{ss}(\omega_i, \mathbf{x}_i, \mathbf{x}_o, \omega_o) = \begin{cases} \frac{1}{\pi r^2} f_s(\omega_i, \mathbf{x}_o, \omega_o) & \text{if } |\mathbf{x}_i - \mathbf{x}_o| < r \\ 0 & \text{else} \end{cases}$$

and then using photon paths to Monte Carlo integrate the corresponding subsurface scattering rendering integral is basically equivalent to photon gathering with a constant disk kernel and the original BSDF f_s. This interpretation allows us to treat mollification as a kernel density estimation in the collapsed dimension.

7.2.2 Vertex Connection and Merging

Vertex Connection and Merging (VCM) [21, 22] combines photon mapping and bidirectional path tracing (BDPT, see section 4.5.4) using MIS. Our regularization technique is based on the derivation by Georgiev et al. [21], although their technique performs merges with multiple photons in the scene simultaneously, whereas we perform merges with a single NEE sample on a light source.

In order to use MIS for combining paths sampled by BDPT and paths based on photon mapping merges, the PDFs for these paths need to be transformed to the same measure. Georgiev et al. treat BDPT paths as usual, and merge vertices $\mathbf{x}_s$ on camera sub-paths $(\mathbf{x}_0, \dots, \mathbf{x}_s, \dots)$ with vertices $\mathbf{x}_s^*$ on light sub-paths $(\dots, \mathbf{x}_s^*, \mathbf{x}_{s+1}, \dots, \mathbf{x}_k)$ as in figure 7.5 right, to construct regular paths $\bar{\mathbf{X}} = (\mathbf{x}_0, \dots, \mathbf{x}_s, \mathbf{x}_{s+1}, \dots, \mathbf{x}_k)$ with PDF

$$p_{VM}^s(\bar{\mathbf{X}}) = P_{acc}^s(\bar{\mathbf{X}}) p_{VC}^s(\bar{\mathbf{X}}),$$

where $p_{VC}^s(\bar{\mathbf{X}})$ is the PDF for sampling $\bar{\mathbf{X}}$ with BDPT by connecting $\mathbf{x}_s$ to $\mathbf{x}_{s+1}$, and $P_{acc}^s(\bar{\mathbf{X}})$ is the acceptance probability for a successful merge between $\mathbf{x}_s$ and $\mathbf{x}_s^*$:

$$\begin{aligned} P_{acc}^s(\bar{\mathbf{X}}) = Pr(\mathbf{x}_s^* \in \mathbb{K}_r(\mathbf{x}_s)) &= \int_{\mathbb{K}_r(\mathbf{x}_s)} p(\mathbf{x}_{s+1} \to \mathbf{x}) \, \mathrm{d}A(\mathbf{x}) \\ &\approx A(\mathbb{K}_r(\mathbf{x}_s)) p(\mathbf{x}_{s+1} \to \mathbf{x}_s^*), \end{aligned}$$

with the exception that P_{acc} is set to 1 if $\mathbf{x}_{s+1}$ is specular.

7.3 Related Work

7.3.1 Regularization for Monte Carlo Rendering

Density estimation. Photon mapping [33] and its extensions, such as progressive photon mapping [34] (a consistent version thanks to an iteratively shrinking kernel) and VCM [21, 22], apply density estimation to approximate the incident radiance at scene vertices. These methods require a preprocessing step to create the photon map and introduce bias everywhere in the image. The following methods and our approach can be applied more selectively to paths that are impossible or hard to sample.

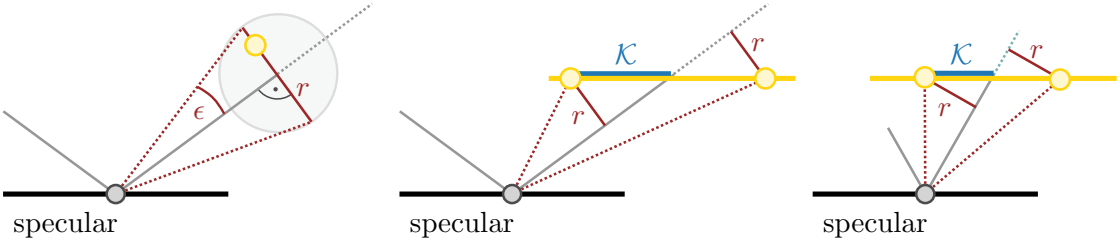


Figure 7.3: Cone mollification accepts an NEE connection if the NEE vertex is within radius r of the true specular direction. This causes non-uniform cone angles ε (center) and kernel shapes ($\mathcal{K}$, blue, compare center and right) on flat area lights.

Cone mollification. Kaplanyan et al. [20] regularize singular-to-singular connections, either between specular surfaces for BDPT paths, or from specular surfaces to point lights for NEE paths, by replacing singular reflectances with a mollified BSDF that is constant within a cone opening angle ε. This angle is determined indirectly to match a sphere of a given radius r centered around where the reflected direction is closest to the connection vertex (see figure 7.3 left). Their method is only applied to perfectly specular connections that are impossible to sample with non-regularized methods, although it can be extended to support area lights. Even so, the resulting kernel shape and area are non-trivial to compute, making this ill-suited for MIS. Our technique works for both impossible and difficult connections, and as our kernel is aligned with the light source allows for more natural MIS weights.

In chapter 6 we apply a similar concept to render spectrally singular BSDFs in the presence of fluorescence.

Roughness regularization. Instead of regularizing the path geometry we can replace specular and near-specular BSDF with a rough microfacet BSDF. The roughness can be based on a threshold [107] or adapted dynamically depending on the path prefix [131].

7.3.2 Regularization Kernels

Previous regularization approaches vary in their choice of kernel shape, profile, size, and shrinking rate. For example, cone regularization [20] and several photon mapping based methods [33, 21, 22] use a uniform disk kernel

$$\mathcal{K}_r(\mathbf{x}|\mathbf{y}) = \begin{cases} \frac{1}{\pi r^2} & \text{if } ||\mathbf{x}-\mathbf{y}|| < r \\ 0 & \text{else.} \end{cases}$$

Their choice for the initial kernel radius is set to a percentage of the scene's bounding box [20, 21], fit to contain a fixed number of photons [33], or picked by hand [22]. Shrinking the radius at a fixed rate ensures consistency [21, 20].

Ray differential tracing [132] and covariance tracing [133] propagate a pixel or light footprint along a path in order to adapt filter kernels at scene vertices, such as for texture filtering [132, 134], to smooth high frequency noise within a pixel footprint [135], or by tracing covariance from the light to adapt the photon kernel radius [136]. We use ray differentials to propagate an (anisotropic) pixel footprint and use it to either construct an anisotropic kernel or to derive a radius for a disk kernel.

7.3.3 Specular Next Event Estimation without Regularization

Manifold sampling approaches iteratively or analytically compute a valid specular vertex [137, 138] or chain [139] connecting the last non-specular vertex of a camera path to a light vertex on point or (small) area lights. Manifold NEE [137] relies on differential geometry and manifold walks to iteratively find such vertices for transmitted caustics. Specular manifold sampling [139] further supports reflections and specular glints, and addresses cases with more than one possible chain between a scene and light vertex.

7.3.4 MIS with Biased Estimators

VCM. Hachisuka et al. [22] derive a bias-aware balance heuristic by minimizing the sum of variance and bias. As the computation of these weights is impractical, they resort to the usual balance heuristic and iterative kernel shrinking via progressive photon mapping, and derive an optimal kernel shrinking rate of 2/3.

BSDF regularization. Spatial cone regularization [20] does not combine biased and unbiased techniques, only picking the regularized technique with the least amount of bias. Microfacet roughness regularization [107, 131] and spectral cone mollification for non-fluorescent BSDFs (chapter 6) use the balance heuristic with fully regularized PDFs.

7.4 Overview

The issues with BSDF sampling and NEE arise if the rendering integral (section 7.2) contains (near) singularities. Our goal is to extend a path tracer with NEE to support connections between specular surfaces and point lights, and to lower the variance of connections between near specular surfaces or small area light sources.

We thus regularize the emitted radiance at point, area and environment lights in section 7.5 to enable NEE from specular surfaces, and apply the same principle to allow BSDF sampling to connect to point lights. Our regularization strategy is akin to the photon merge, but only uses one light sample and takes place directly at the light source. We adapt the regularization kernel shape and size to the pixel footprint in section 7.6.

Our path tracer is outlined in algorithm 4 and figure 7.4, with the changes to a conventional path tracer marked in yellow. Figure 7.5 shows the geometric setup of our regularization for an area light after a singular surface. We sample a vertex $\mathbf{x}_{nee}$ on an emitter just as we would for normal NEE. We then construct a vertex $\mathbf{x}^*$ on the (near) specular direction ω_s from BSDF sampling as described in section 7.5.1, and center the regularization kernel on that vertex. If the NEE sample $\mathbf{x}_{nee}$ is within this kernel we accept the connection and accumulate the path throughput weighted with the kernel.

Just like normal NEE, our technique comes with its own variance. In cases that BSDF sampling handles well, this variance may even be higher than the unregularized BSDF sampling variance. This generally happens whenever the light source is larger than the kernel, as in the right columns in figure 7.6, where either the kernel is small (left grid) or the light source is large (right grid). We thus combine samples from both techniques using MIS to minimize the overall variance in sections 7.7 and 7.8.

In section 7.9 we evaluate our technique and weights for specular BSDFs and area lights with the path tracer from algorithm 4.

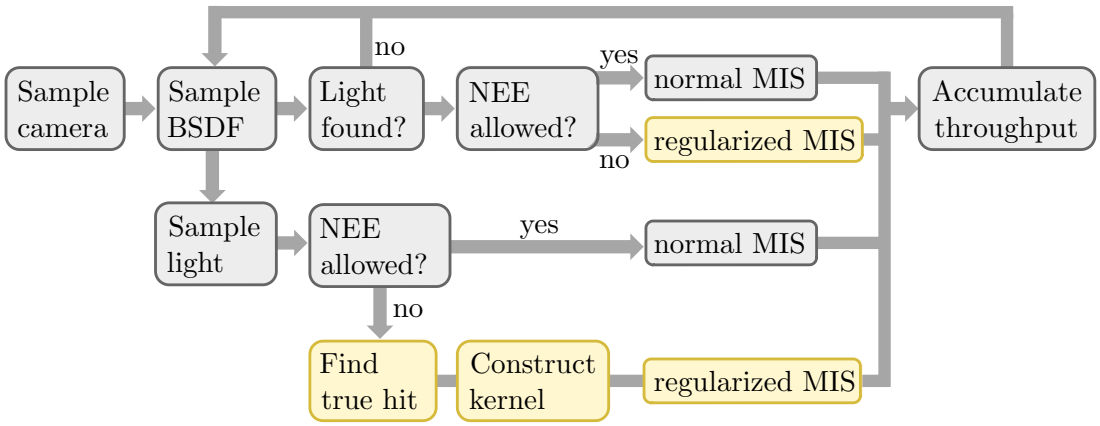


Figure 7.4: We extend a typical path tracing flow by allowing NEE from specular scene vertices. We accept NEE vertices within a kernel around the "true" hit where the specular direction intersects the light source plane, and we adapt the MIS weights accordingly. New components are marked yellow.

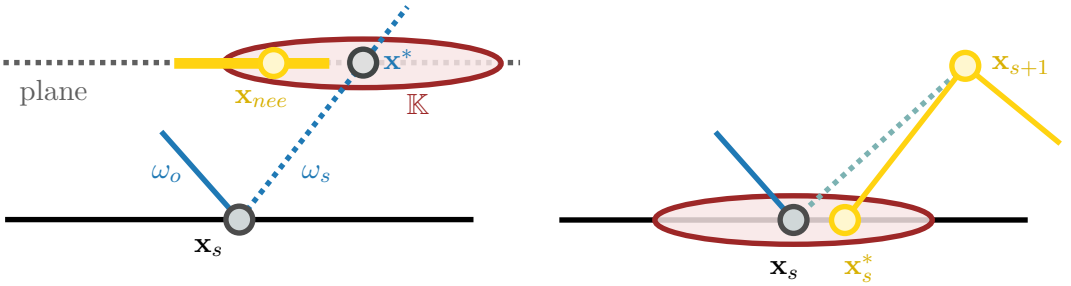


Figure 7.5: **Left:** We accept connections if the NEE vertex $\mathbf{x}_{nee}$ is within a kernel $\mathbb{K}$ centered at the true hit vertex $\mathbf{x}^*$, where the specular direction ω_s intersects a plane aligned with the light source at $\mathbf{x}_{nee}$. **Right:** VCM accepts a photon at $\mathbf{x}_s^*$ if it is within a disk kernel centered around the last vertex $\mathbf{x}_s$ of a camera sub-path.

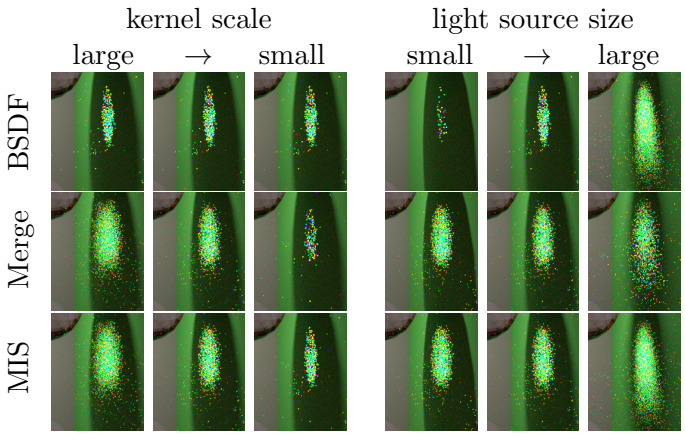


Figure 7.6: Our MIS weights favor the technique with the lower variance, i.e. our technique if the kernel is large or the light source is small (left columns), and BSDF sampling if the kernel is small or the light source is large (right columns).

Algorithm 4: Forward path tracer with regularized NEE

```
x_0 ← sample camera
x_1 ← intersect camera ray with scene
for s = 1; s < max length; s++ do
    // connect x_s to a light vertex
    x_nee ← sample NEE vertex
    if x_s specular then
        i ← sample specular component at x_s
        ω_i ← deterministic direction for component i
        x* ← intersect ω_i with plane around x_nee
        determine regularization kernel at x*
        if x_nee ∈ K_r(x*) then
            w_merge = regularized MIS weight
            accumulate w_merge· throughput ·K_r
    else
        w_nee = normal NEE MIS weight
        accumulate w_nee· throughput
    // continue the path
    x_{s+1} ← BSDF sampling at x_s
    if x_{s+1} on light then
        if x_s specular then
            w_BSDF = regularized MIS weight
        else
            w_BSDF = normal MIS weight
        accumulate w_BSDF· throughput and quit
```

7.5 Merge-Based NEE Regularization

In this section we present our framework to regularize next event estimation (NEE) for connections from non-singular lights to surfaces (section 7.5.1), and then extend it to cover connections involving one (section 7.5.2) or two simultaneous (section 7.5.3) singularities.

Our explanation assumes that a camera path prefix $(\mathbf{x}_0, \ldots, \mathbf{x}_s)$ has been sampled with $\omega_o = \mathbf{x}_s \to \mathbf{x}_{s-1}$ as the outgoing direction at the last scene vertex $\mathbf{x}_s$ (see figure 7.5). Since we only regularize the last step during path construction, this section only considers the direct illumination at $\mathbf{x}_s$.

7.5.1 NEE Merging without Singularities

The outgoing radiance due to direct illumination at $\mathbf{x}_s$ along ω_o is

$$L_o(\mathbf{x}_s, \omega_o) = \int_M L_e(\mathbf{x}' \to \mathbf{x}_s) f_s(\mathbf{x}', \mathbf{x}_s, \mathbf{x}_{s-1}) G(\mathbf{x}', \mathbf{x}_s) \, \mathrm{d}A(\mathbf{x}'). \tag{7.5.1}$$

We can estimate L_o with Monte Carlo integration by sampling a vertex $\mathbf{x}^*$ from the BSDF at $\mathbf{x}_s$ with density $p_A(\mathbf{x}^*)$:

$$L_o(\mathbf{x}_s, \omega_o) \approx \frac{L_e(\mathbf{x}^* \to \mathbf{x}_s) f_s(\mathbf{x}^*, \mathbf{x}_s, \mathbf{x}_{s-1}) G(\mathbf{x}^*, \mathbf{x}_s)}{p_A(\mathbf{x}^*)}. \tag{7.5.2}$$

Similar to VCM [21], we use density estimation with a uniform 2D kernel $\mathcal{K}_r = 1/A(\mathbb{K}_r)$ with support $\mathbb{K}_r = supp(\mathcal{K}_r)$ to approximate $L_e(\mathbf{x}^* \to \mathbf{x}_s)$

$$L_e(\mathbf{x}^* \to \mathbf{x}_s) \approx \frac{1}{A(\mathbb{K}_r)} \int_{\mathbb{K}_r(\mathbf{x}^*)} L_e(\mathbf{x}'' \to \mathbf{x}_s)\, \mathrm{d}A(\mathbf{x}'')$$

with the average outgoing radiance over the kernel support $\mathbb{K}_r$. The kernel support $\mathbb{K}_r(\mathbf{x}^*)$ is centered at $\mathbf{x}^*$, has the same normal n^* as the surface at $\mathbf{x}^*$, and has scale r, i.e. $A(\mathbb{K}_r) \propto r^2$ (see figure 7.5). Next, we perform an additional Monte Carlo estimate using one sample $\mathbf{x}_{nee}$ drawn from a density p_A^{nee} for this integral

$$\int_{\mathbb{K}_r(\mathbf{x}^*)} L_e(\mathbf{x}'' \to \mathbf{x}_s)\, \mathrm{d}A(\mathbf{x}'') \approx \frac{\chi_{\mathbb{K}_r(\mathbf{x}^*)}(\mathbf{x}_{nee}) L_e(\mathbf{x}_{nee} \to \mathbf{x}_s)}{p_A^{nee}(\mathbf{x}_{nee})}, \tag{7.5.3}$$

where $\chi_{\mathbb{K}_r(\mathbf{x}^*)}$ is the kernel's characteristic function

$$\chi_{\mathbb{K}_r(\mathbf{x}^*)}(\mathbf{x}) = \begin{cases} 1 & \text{if } x \in \mathbb{K}_r(\mathbf{x}^*) \\ 0 & \text{else.} \end{cases}$$

Note that p_A^{nee} does not need to be restricted to the kernel support, although we get additional bias if p_A^{nee} does not cover the full kernel. Our merge between $\mathbf{x}^*$ and $\mathbf{x}_{nee}$ at the light source is similar to the VCM vertex merge at the last camera sub-path vertex $\mathbf{x}_s$. The main difference is that we shift the merge location from the last vertex on a path prefix to the NEE vertex at the light source.

Combining these steps with equation 7.5.2 results in the biased estimator

$$L_o(\mathbf{x}_s, \omega_o) \approx \frac{\chi_{\mathbb{K}_r(\mathbf{x}^*)}(\mathbf{x}_{nee}) L_e(\mathbf{x}_{nee} \to \mathbf{x}_s) f_s(\mathbf{x}^*, \mathbf{x}_s, \mathbf{x}_{s-1}) G(\mathbf{x}^*, \mathbf{x}_s)}{A(\mathbb{K}_r) p_A^{nee}(\mathbf{x}_{nee}) p_A(\mathbf{x}^*)}. \tag{7.5.4}$$

The bias effectively makes some effects appear as if the scene had been lit by larger light sources. We later extend this concept to also sample paths where either the surface at $\mathbf{x}_s$ is perfectly specular or the light source is a point light (section 7.5.2), or both (section 7.5.3).

Sampling procedure. We sample $\mathbf{x}^*$ with BSDF sampling and $\mathbf{x}_{nee}$ as we would usually sample a vertex for next event estimation. We start by sampling a direction ω_{or} from the BSDF with density $p_\sigma^{BSDF}(\omega_{or})$ in solid angle measure, and an NEE vertex $\mathbf{x}_{nee}$ from a density p_A^{nee} in surface area measure. Next we intersect ω_{or} with a virtual plane around $\mathbf{x}_{nee}$ to get $\mathbf{x}^*$, and convert $p_\sigma(\omega_{or})$ to $p_A(\mathbf{x}^*)$. The plane's normal is the same as the normal at $\mathbf{x}_{nee}$ if it is on an area light (figure 7.5), and ω_{or} if $\mathbf{x}_{nee}$ is a point light (same as figure 7.3 left). We decided against using ω_{or} as the virtual plane's normal for area lights, as this distorts the regularization effect (figure 7.15, figure 7.3 right) and makes the derivation of MIS weights (section 7.8) less intuitive.

Fixed position or fixed direction? We keep $\mathbf{x}_s$ fixed in the density estimate for $L_e(\mathbf{x}^* \to \mathbf{x}_s)$, which means that the outgoing radiance at $\mathbf{x}_{nee}$ is evaluated for a different direction $(\mathbf{x}_{nee} \to \mathbf{x}_s)$ than the BSDF $(\omega_i = \mathbf{x}^* \to \mathbf{x}_s)$. Instead we could have kept the BSDF direction ω_i fixed, using the estimate

$$L_e(\mathbf{x}^*, \omega_i) \approx 1/A(\mathbb{K}_r) \int_{\mathbb{K}_r(\mathbf{x}^*)} L_e(\mathbf{x}_{nee}, \omega_i)\, \mathrm{d}A(\mathbf{x}_{nee}).$$

Since the only difference is for which direction we evaluate the emitted radiance at the light, we expect similar results for both and simply chose the first option because it fits better in our renderer's code. Both choices converge to the same value in the limit for $\mathbb{K}_r(\mathbf{x}^*) \to \{\mathbf{x}^*\}$ as $r \to 0$, and the reasoning in the remainder of this paper works for both options. Similarly, we can evaluate visibility for $\mathbf{x}^*$ or $\mathbf{x}_{nee}$, and we compare the two visibility options in section 7.9.3.

7.5.2 NEE Merging with One Singularity

If the light source or BSDF contains a singularity, we cannot start our derivations from equation 7.5.1, since the outgoing radiance at $\mathbf{x}_s$ is no longer described by an integral. In this section, we build on the equations for outgoing radiance in the presence of a single singularity to derive estimators corresponding to equation 7.5.4.

Case 1: Singular BSDF and continuous emitter. If the surface at $\mathbf{x}_s$ has N singular components (e.g. $N = 2$ for reflect and transmit) with albedos $a_1, \dots, a_N$, we replace the rendering integral with

$$L_o(\mathbf{x}_s, \omega_o) = \sum_{i=1}^{N} L_i(\mathbf{x}_s, \omega_i) a_i. \tag{7.5.5}$$

As in normal path tracing, we sample one of the singular components with probability $P_s(i)$ and compute its deterministic direction ω_i. We regularize the incident light with a kernel

$$L_i(\mathbf{x}_s, \omega_i) = L_e(\mathbf{x}^* \to \mathbf{x}_s) \approx \frac{1}{A(\mathbb{K}_r)} \int_{\mathbb{K}_r(\mathbf{x}^*)} L_e(\mathbf{x}' \to \mathbf{x}_s) \, \mathrm{d}A(\mathbf{x}'), \tag{7.5.6}$$

where $\mathbf{x}^*$ is constructed by intersecting ω_i with the plane around the NEE vertex $\mathbf{x}_{nee}$ sampled from p_A^{nee}. To estimate $L_o(\mathbf{x}_s, \omega_o)$ we replace the continuous density $p_A(\mathbf{x}^*)$ in equation 7.5.4 with the probability $P_s(i)$ for selecting the i-th singular component:

$$L_o(\mathbf{x}_s, \omega_o) \approx \frac{\chi_{\mathbb{K}_r}(\mathbf{x}_{nee}) L_e(\mathbf{x}_{nee} \to \mathbf{x}_s) a_i}{A(\mathbb{K}_r) p_A(\mathbf{x}_{nee}) P_s(i)}. \tag{7.5.7}$$

In general, if we have a specular chain $(\mathbf{x}_{d+1}, \dots, \mathbf{x}_s)$ instead of a single specular vertex, this extends to

$$L_o(\mathbf{x}_{d+1}, \omega_o) \approx \frac{\chi_{\mathbb{K}_r}(\mathbf{x}_{nee}) L_e(\mathbf{x}_{nee} \to \mathbf{x}_s) \prod_{j=d+1}^{s} a_{i_j}}{A(\mathbb{K}_r) p_A(\mathbf{x}_{nee}) \prod_{j=d+1}^{s} P_j(i_j)},$$

where we accumulate the singular albedos in $\prod a_{i_j}$ and the specular component selection probabilities P_j along the chain.

For environment maps we could sample ω_{nee} from the environment map in solid angle density, and use it to estimate the radiance within a kernel centered around ω_i in the environment map. In practice we replace this with a prefiltered environment map evaluation to reduce variance (see appendix C.2).

Case 2: Continuous BSDF and singular emitter(s). For N point light sources at positions $\mathbf{x}_j$ with intensities I_j we replace the rendering integral with

$$L_o(\mathbf{x}_s, \omega_o) = \sum_{j=1}^{N} I_j f_s(\mathbf{x}_j \to \mathbf{x}_s, \omega_o) G'(\mathbf{x}_j, \mathbf{x}_s). \tag{7.5.8}$$

The geometry term G' lacks the cosine at $\mathbf{x}_j$, since $\mathbf{x}_j$ is a point without a normal. We sample the j-th point light with probability $P_{PL}(j)$, resulting in the light vertex $\mathbf{x}_{nee} := \mathbf{x}_j$. To unify this with the framework we have so far, we replace $f_s(\mathbf{x}_{nee} \to \mathbf{x}_s, \omega_o)$ with a kernel density estimate

$$f_s(\mathbf{x}_{nee} \to \mathbf{x}_s, \omega_o) \approx \frac{1}{A(\mathbb{K}_r)} \int_{\mathbb{K}_r(\mathbf{x}_{nee})} f_s(\mathbf{x} \to \mathbf{x}_s, \omega_o) \, \mathrm{d}A(\mathbf{x})$$

over a kernel centered around $\mathbf{x}_{nee}$ and perpendicular to $\mathbf{x}_{nee} \to \mathbf{x}_s$. We use Monte Carlo integration to estimate this as

$$\int_{\mathbb{K}_r(\mathbf{x}_{nee})} f_s(\mathbf{x} \to \mathbf{x}_s, \omega_o) \, \mathrm{d}A(\mathbf{x}) \approx \frac{\chi_{\mathbb{K}_r(\mathbf{x}^*)}(\mathbf{x}_{nee}) f_s(\mathbf{x}^* \to \mathbf{x}_s, \omega_o)}{p_A^{BSDF}(\mathbf{x}^*)}.$$

Again, we construct $\mathbf{x}^*$ by sampling a direction from the BSDF and intersecting it with the kernel plane. In the resulting estimator for $L_o(\mathbf{x}_s, \omega_o)$ the continuous density p_A^{nee} from equation 7.5.4 is replaced by the probability $P_{PL}(j)$ for selecting the j-th point light:

$$L_o(\mathbf{x}_s, \omega_o) \approx \frac{\chi_{\mathbb{K}_r(\mathbf{x}^*)}(\mathbf{x}_{nee}) I_j f_s(\mathbf{x}^* \to \mathbf{x}_s, \omega_o) G'(\mathbf{x}_{nee}, \mathbf{x}_s)}{A(\mathbb{K}_r) P_{PL}(j) p_A^{BSDF}(\mathbf{x}^*)}.$$

Compared to equation 7.5.4, the BSDF PDF for $\mathbf{x}^*$ is in surface area measure for the kernel plane around $\mathbf{x}_{nee}$, and a conversion to solid angle measure no longer cancels out with the geometry term, which now depends on $\mathbf{x}_{nee}$ instead of $\mathbf{x}^*$ and lacks the cosine at $\mathbf{x}_{nee}$.

7.5.3 NEE Merging with Two Singularities

For simplicity, this section derives our merge technique for a single singular surface with one mirror component with a single point light. The reasoning trivially extends to surfaces with multiple specular components or point lights by summing up the components as in equations 7.5.5 and 7.5.8, and importance sampling that sum with a discrete selection probability to select a specular component (see section 7.5.4) and point light. The intuitions for specular chains and refractions leading up to the point light are the same, but the notation is more extensive, so we show the derivation in appendix C.1 and present the resulting estimator at the end of this section in equation 7.5.11.

One mirror, one point light. Due to the singular reflectance at $\mathbf{x}_s$ and the singular light source at $\mathbf{x}_p$ as in figure 7.7 (left), we can no longer describe the outgoing radiance at the path prefix's *previous* vertex $\mathbf{x}_{s-1}$ with the typical rendering integral

$$L_o(\mathbf{x}_{s-1} \to \mathbf{x}_{s-2}) = \int_M L_o(\mathbf{x}_s, \mathbf{x}_{s-1}) f_s(\mathbf{x}_{s-1}) G(\mathbf{x}_s, \mathbf{x}_{s-1}) \, \mathrm{d}A(\mathbf{x}_s).$$

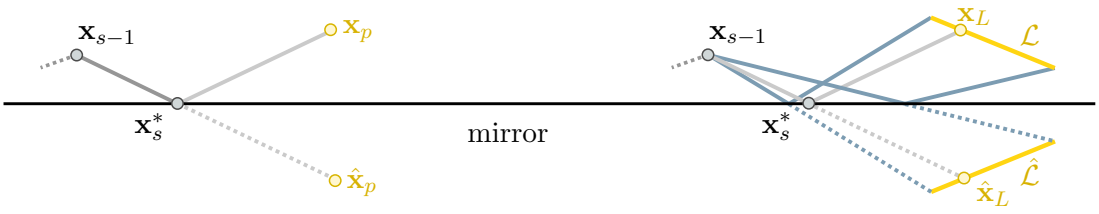


Figure 7.7: If a point light at $\mathbf{x}_p$ (**left**) or an area light $\mathcal{L}$ (**right**) is reflected at $\mathbf{x}_s^*$, it appears to be at $\hat{\mathbf{x}}_p$ or $\hat{\mathcal{L}}$ when viewed from $\mathbf{x}_{s-1}$.

This is because there exists no function for radiance[1] $L_o(\mathbf{x}_s \to \mathbf{x}_{s-1})$ that we could put in the integrand for the actual behavior

$$L_o(\mathbf{x}_{s-1} \to \mathbf{x}_{s-2}) = f_s(\mathbf{x}_s^* \to \mathbf{x}_{s-1} \to \mathbf{x}_{s-2}) E(\mathbf{x}_{s-1}), \tag{7.5.9}$$

where $\mathbf{x}_s^*$ is the position on the singular surface where the point light is visible from $\mathbf{x}_{s-1}$ (see figure 7.7 left). $E(\mathbf{x}_{s-1})$ is the irradiance at $\mathbf{x}_{s-1}$ caused by the reflected point light, which for a point light of intensity I_p and a mirror at $\mathbf{x}_s$ is

$$E(\mathbf{x}_{s-1}) = \frac{I_p \cos_{\mathbf{x}_{s-1}}(\mathbf{x}_{s-1} \to \mathbf{x}_s^*)}{(||\mathbf{x}_s^* - \mathbf{x}_{s-1}|| + ||\mathbf{x}_p - \mathbf{x}_s^*||)^2}.$$

This means we can rewrite $L_o(\mathbf{x}_{s-1})$ as

$$L_o(\mathbf{x}_{s-1} \to \mathbf{x}_{s-2}) = I_p f_s(\mathbf{x}_s^* \to \mathbf{x}_{s-1} \to \mathbf{x}_{s-2}) G'(\hat{\mathbf{x}}_p, \mathbf{x}_{s-1}), \tag{7.5.10}$$

taking the geometry term between $\mathbf{x}_{s-1}$ and the mirrored position $\hat{\mathbf{x}}_p$ where the point light appears to be when viewed from $\mathbf{x}_{s-1}$. In particular, we do not need two separate geometry terms $G(\mathbf{x}_{s-1}, \mathbf{x}_s)$ and $G(\mathbf{x}_s, \mathbf{x}_p)$. In appendix C.1 we show how to derive this equation as the limit of a shrinking sphere light and specular surface, and discuss the pitfalls when trying to do so using Dirac δ-abbreviations.

Similar to Case 2, we can now smooth out the emitted light, effectively replacing the point light at $\mathbf{x}_p$ with a kernel-shaped area light centered around $\mathbf{x}_p$ and facing $\mathbf{x}_s^*$:

$$L_o(\mathbf{x}_{s-1}, \mathbf{x}_{s-2}) \approx \int_{\mathbb{K}_r(\mathbf{x}_p)} \frac{I_p}{A(\mathbb{K}_r)} f_s(\mathbf{x}_s^{**}, \mathbf{x}_{s-1}, \mathbf{x}_{s-2}) G(\hat{\mathbf{x}}_p', \mathbf{x}_{s-1}) \, \mathrm{d}A(\mathbf{x}_p'),$$

with $\mathbf{x}_s^{**}$ as the position on the mirror where we see $\mathbf{x}_p'$ on the kernel-shaped light from $\mathbf{x}_{s-1}$ (just like $\mathbf{x}_s^*$ and $\mathbf{x}_L$ in figure 7.7 right).

We estimate this by sampling a direction ω_i from the BSDF at $\mathbf{x}_{s-1}$ and tracing it to intersect the plane around the light to get $\mathbf{x}_p'$. Parts of the geometry term $G(\mathbf{x}_p', \mathbf{x}_{s-1})$ cancel out with the BSDF PDF converted to surface area measure around $\mathbf{x}_p$, leaving only the cosine $\cos_{\mathbf{x}_{s-1}}(\mathbf{x}_{s-1} \to \mathbf{x}_p')$ and solid angle PDF p_σ. If $\mathbf{x}_p'$ is inside the kernel around $\mathbf{x}_p$ we get a contributing path

$$L_o(\mathbf{x}_{s-1} \to \mathbf{x}_{s-2}) \approx \begin{cases} \frac{1}{A(\mathbb{K}_r)} \frac{I_p f_s \cos(\omega_i)}{p_\sigma(\omega_i)} & \text{if } \mathbf{x}_p' \in \mathbb{K}_r(\mathbf{x}_p) \\ 0 & \text{else.} \end{cases}$$

[1] Another interpretation is that there is no function for the exitant radiance at the mirror in the first place, as radiance is defined as the derivative of power per area per solid angle, but since the power at the mirror is 0 everywhere except in the single reflected direction $r(\omega_p)$, this derivative does not exist at $r(\omega_p)$ and is 0 everywhere else.

The expected value of this converges to equation 7.5.9 in the limit for $r \to 0$, because of equation 4.8.4, similar to the derivation in appendix C.1.1.

Alternatively we could sample this by directly picking a sample $\mathbf{x}'_p$ on the kernel light, but tracing it back to compute the intermediate path vertices between $\mathbf{x}_{s-1}$ and $\mathbf{x}'_p$ is not trivial, especially for longer specular chains, and requires specialized approaches (see section 7.3.3). In fact, if this was easy we could do so for $\mathbf{x}_p$ directly without regularization.

Specular chains and refractions. Extending this concept to longer specular chains and/or refractions (see appendix C.1) results in the estimator

$$L_o(\mathbf{x}_d, \mathbf{x}_{d-1}) \approx \begin{cases} \frac{1}{A(\mathbb{K}_r)} \frac{I_p f_s \prod a_{i_j} \cos_{\mathbf{x}_d}(\omega)}{p_\sigma(\omega) \prod P_j(i_j)} & \text{if } \mathbf{x}_{nee} \in \mathbb{K}_r(\mathbf{x}_p) \\ 0 & \text{else.} \end{cases} \tag{7.5.11}$$

In particular, we now divide by the product of all selection probabilities of the specular components along the specular chain.

7.5.4 Forcing the BSDF Hemisphere

With independent BSDF and NEE sampling, whenever the BSDF supports both hemispheres, regularization always fails if the BSDF sample at $\mathbf{x}_s$ is on the opposite hemisphere of the NEE sample. We can avoid this by forcing the BSDF sampling procedure to the NEE sample's hemisphere. We thus split the BSDF PDF or selection probability

$$p_s = \mathbb{P}_s^+ p_s^+ + \mathbb{P}_s^- p_s^-$$

into PDFs or selection probabilities p_s^+, p_s^- conditionalized to the upper and lower hemisphere, weighted with probabilities $\mathbb{P}_s^+$ and $\mathbb{P}_s^- = 1 - \mathbb{P}_s^+$ for choosing the upper or lower hemisphere at $\mathbf{x}_s$.

We then use two regularized estimators, one with an upper-hemisphere sample from p_s^+, and one with a lower-hemisphere sample from p_s^-, and correlate them by using the same NEE sample for both. Since one of the two estimators is always zero, in practice we only need to draw a sample in the NEE sample's hemisphere, and divide the estimators not by p_s, but instead by p_s^+ for upper- and p_s^- for lower-hemisphere samples, respectively.

7.6 Adaptive Regularization Kernels

Our approach to regularize the emitted radiace for (near-)specular NEE works with any 2D kernel that can be easily evaluated for a single vertex, although some of our MIS weights (section 7.8) further assume that the kernel is uniform, i.e. $\mathcal{K}_r \equiv 1/A(\mathbb{K}_r)$. We test a disk kernel and kernels based on a (potentially anisotropic) pixel footprint. In both cases we scale the kernel with a shrinking r as required by equation 4.8.5 in order to achieve a consistent result, although we also show renders with a constant kernel scale to better highlight the resulting bias and the effect of MIS.

In order to adapt the kernel size to the path we propagate the pixel footprint of a pixel position p by tracing ray differentials [132] from $p + (1, 0)$ and $p + (0, 1)$. For scene lights we use the positional differentials to construct one of the kernel shapes in section 7.6.1. For environment maps we use the directional differentials to prefilter the environment map directly with an isotropic or anisotropic lookup, as discussed in appendix C.2.

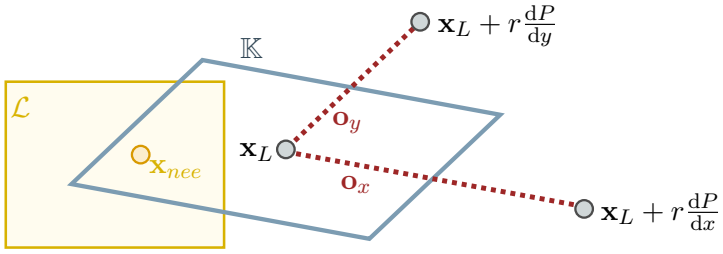


Figure 7.8: We accept an NEE connection if the NEE vertex is within the parallelogram $\mathbb{K}$ of the scaled positional ray differentials $\mathbf{o}_x = r \cdot \mathrm{d}P/\,\mathrm{d}x$ and $\mathbf{o}_y = r \cdot \mathrm{d}P/\,\mathrm{d}y$ in the light source plane.

7.6.1 Kernel Shapes for Point or Area Lights

If the NEE vertex is on a light source in the scene, we place the 2D kernel in a plane that is either orthogonal to the connecting direction toward a point light, or orthogonal to the light source normal at the NEE vertex. We then use the positional ray differentials in this plane to construct the kernel. We tried three kernel shapes for scene light sources: Disks, parallelograms (figure 7.8), and ellipses.

Disk kernel. As the ray differentials usually have different extents, we need to derive a radius for an isotropic disk kernel from an anisotropic footprint. For simplicity we either pick the average, minimum or maximum length of the two positional differentials.

Parallelogram kernel. We construct a uniform parallelogram-shaped kernel based on the positional differentials as shown in figure 7.8, where we accept the merge if the NEE vertex is within the parallelogram. Given differential vectors $\mathbf{o}_x, \mathbf{o}_y$, the parallelogram kernel's area is $A(\mathbb{K}_r) = 4 \cdot ||\mathbf{o}_x \times \mathbf{o}_y|| = 4 \cdot |\mathbf{o}_x| \cdot |\mathbf{o}_y| \cdot \sin(\mathbf{o}_x, \mathbf{o}_y)$.

Elliptical kernel. Finally, we construct an elliptical kernel containing the two positional differential vectors. Since the differentials are not necessarily orthogonal, we use the longer one as the ellipse's major axis, and compute the minor axis such that the smaller differential connects the ellipse's center to its border.

7.6.2 Kernel Scale

The positional ray differentials only provide the propagated pixel footprint. Basing the kernel directly on the differentials thus restricts the density estimation to the pixel footprint, causing no noticeable bias but strong noise. If we want to trade noise for bias, we need to scale the footprint-based kernel by a user parameter. In case of connections to a scene vertex this is rather trivial as we can simply scale the 2D kernel, which is equivalent to scaling the positional differential vectors before constructing the kernel. In case of connections to an environment map, we scale the directional ray differentials and construct the kernel from the scaled differentials in texture space. The resulting opening angle only changes close to linearly if the scaled differential is small enough, but this was not an issue in any of our experiments.

7.7 Multiple Importance Sampling: VCM-Style

In this section we follow the derivations by Georgiev et al. [21] for a balance heuristic to combine our regularized NEE estimators from equation 7.5.4 or 7.5.7 with (discrete) BSDF sampling for connections from (near) specular surfaces to area light sources.

Acceptance probability. The probability of an NEE sample $\mathbf{x}_{nee}$ with density p_{nee} falling within the kernel around $\mathbf{x}^*$ is

$$P_{accept} = \int_{\mathbb{K}_r(\mathbf{x}^*)} p_A^{nee} \, \mathrm{d}A(\mathbf{x}) \approx p_{nee}(\mathbf{x}_{nee}) \cdot A(\mathbb{K}_r) = \frac{A(\mathbb{K}_r)}{A(\mathcal{L})}. \tag{7.7.1}$$

This differs from the acceptance probability used in VCM [21] in that it no longer depends on a PDF for the BSDF at the vertex leading up to the merge vertex, but instead a PDF for explicit light source sampling. In particular, if that BSDF is singular, we no longer need to set the acceptance probability to 1.

If the light source $\mathcal{L}$ is smaller than the kernel, the approximated "probability" becomes greater than one, but this is not an issue: In the estimator itself, this cancels out the bias from the Monte Carlo kernel density estimation (equations 7.5.3 and 7.5.6) with a PDF p_A^{nee} that is 0 for parts of the kernel support. Meanwhile, the MIS weight increases with growing kernels, reflecting that the variance of the regularized estimator decreases with growing kernel sizes (see section 7.8.1).

The approximation breaks if multiple light sources are present or if the light source is singular, as we discuss at the end of this section. We use the approximation for multiple emitters anyway, and we do not support MIS with point lights since we cannot sample them from the BSDF in the first place.

MIS weights. The densities for the light vertex on merged paths and BSDF-sampled paths are

$$\begin{aligned} p_\sigma^{merge}(\mathbf{x}^*|\mathbf{x}_0, \dots, \mathbf{x}_s) &= P_{accept} \cdot p_\sigma(\mathbf{x}_s \to \mathbf{x}^*) \\ p_\sigma^{BSDF}(\mathbf{x}^*|\mathbf{x}_0, \dots, \mathbf{x}_s) &= p_\sigma(\mathbf{x}_s \to \mathbf{x}^*). \end{aligned}$$

If the BSDF at $\mathbf{x}_s$ is singular, p_σ is replaced by the probability for selecting the specular component $\mathbf{x}_s \to \mathbf{x}^*$, e.g. (one minus) the Fresnel term if $\mathbf{x}_s$ is a dielectric. Either way, p_σ fully cancels out in the balance heuristic (equation 4.3.5) with $p_\sigma^{merge}, p_\sigma^{BSDF}$:

$$\begin{aligned} w_{BSDF}(\mathbf{x}^*|\mathbf{x}_0, \dots, \mathbf{x}_s) &= \frac{1}{1 + P_{accept}} \\ w_{merge}(\mathbf{x}^*|\mathbf{x}_0, \dots, \mathbf{x}_s) &= \frac{P_{accept}}{1 + P_{accept}}. \end{aligned}$$

Since $P_{accept} \propto A(\mathbb{K}_r)$, the MIS weight w_{merge} for the regularized technique is smaller for paths with a smaller kernel scale (and thus higher variance). Furthermore, since $P_{accept} \propto p_A(\mathbf{x}_{nee})$, if $p_A(\mathbf{x}_{nee}) \propto 1/A(\mathcal{L})$ the weight is larger for smaller light sources, where regularization is likely more important than at larger lights.

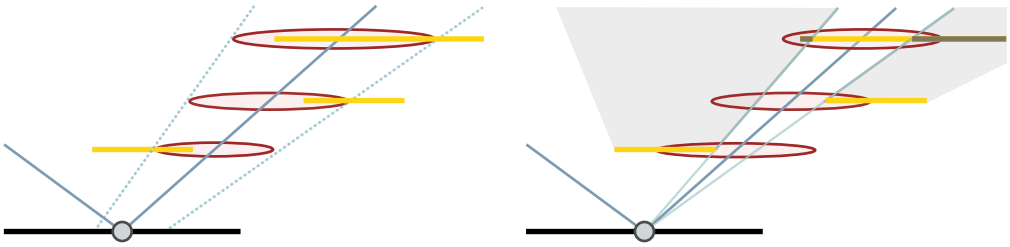


Figure 7.9: **Left**: Three light sources at different distances result in different kernel sizes based on the ray differentials. **Right**: The approximation for the acceptance probability remains reasonable if we only accept visible NEE vertices with a constant kernel size.

Forcing the BSDF hemisphere. If we replace the merge strategy with an upper- and lower-hemisphere merge at the last specular vertex as in section 7.5.4, the two regularized strategies correspond to

$$p_\sigma^{merge+} = p_\sigma^+(\omega_i) \cdot P_{accept} \qquad p_\sigma^{merge-} = p_\sigma^-(\omega_i) \cdot P_{accept},$$

where $p^{+,-}$ is either a PDF for a continuous BSDF or a probability function for a singular BSDF at $\mathbf{x}_s$, conditionalized to the upper and lower hemisphere, respectively. Since the BSDF sampling strategy still samples the full sphere, $p_\sigma = \mathbb{P}^+ p_\sigma^+ + \mathbb{P}^- p_\sigma^-$ no longer fully cancels out and we get the MIS weights

$$\begin{aligned} w_{merge}^+ &= \frac{P_{accept}}{P_{accept} + \mathbb{P}^+} \qquad w_{merge}^- = \frac{P_{accept}}{P_{accept} + \mathbb{P}^-} \\ w_{BSDF} &= \begin{cases} \frac{\mathbb{P}^+}{P_{accept}+\mathbb{P}^+} & \text{if mode} = \text{reflect} \\ \frac{\mathbb{P}^-}{P_{accept}+\mathbb{P}^-} & \text{if mode} = \text{transmit.} \end{cases} \end{aligned} \tag{7.7.2}$$

MIS with multiple light sources. In scenes with multiple light sources the acceptance probability (equation 7.7.1) changes, since NEE vertices on different light sources produce different kernel centers and scales (see Fig. 7.9). For instance, consider a ray with many nearby light sources $\mathcal{L}_1, \ldots, \mathcal{L}_l$, each of them containing NEE samples that contribute to our regularized technique through kernels placed on corresponding points $\mathbf{x}_L$ along the ray. Computing the overall probability of a successful regularization now requires a summation over all light sources:

$$P_{accept} = \sum_{\mathcal{L}_i} \int_{\mathbb{K}_r(\mathbf{x}_L)} p_A^{nee}(\mathbf{x}) \, \mathrm{d}A(\mathbf{x}) \approx p_A^{nee}(\mathbf{x}_{nee}) \cdot \sum_{\mathcal{L}_i} A(\mathbb{K}_r(\mathbf{x}_L)).$$

One fix for this issue is to always use a fixed kernel radius and only accept regularization if the NEE sample is visible, thus splitting up the kernel and placing parts of it at different light sources, in which case $P_{accept} = p_A^{nee} A(\mathbb{K})$ is appropriate. However, this no longer works for adaptive kernel radii, where the overall area of the split kernel no longer equals the area of the base kernel. We therefore use $P_{accept} \approx p_A^{nee}(\mathbf{x}_{nee}) A(\mathbb{K}_r)$ anyway and leave this for future work.

Acceptance probability for point lights. If the emitter itself is singular, the acceptance probability no longer applies. For instance, if we view the singular emitter as the limit of a series of shrinking area lights, then $p_{nee} \to \infty$ and thus $P_{accept} \to \infty$ as well. This requires a different point of view, e.g. shifting the acceptance probability from the light PDF to the BSDF PDF. We leave this open to future work and only investigate MIS with finite light sources, since point lights cannot be sampled by BSDF sampling anyway. In other words, there is no BSDF sampling strategy with which to combine our merge estimator in the first place.

7.8 Variance-Minimizing MIS Weights

In this section we discuss alternatives to the VCM-style MIS weights (equation 7.7.2). First, we analyze the individual variance of the BSDF-sampling and regularized estimator (section 7.8.1). Next, we propose naive heuristic weights using the kernel and light source area in the balance heuristic (section 7.8.2) that effectively result in the same weighting functions as equation 7.7.2. We then derive new MIS weights to minimize the variance of the combined estimator (section 7.8.3), analyze how their behavior in the limit relates to the VCM-style weights (section 7.8.4) and discuss heuristic approximations of these weights (section 7.8.5). Lastly, we discuss what changes if we force the BSDF to sample the hemisphere of the NEE sample (section 7.8.6), and if there is more than one emitter in the scene (section 7.8.7).

7.8.1 Individual Variance Analysis

In this section we analyze the variance

$$Var(F) = \mathbb{E}(F^2) - \mathbb{E}^2(F)$$

of the BSDF estimator F_{BSDF} and the regularized estimator F_{merge} for $L_o(\mathbf{x}_d, \omega_o)$ where light arrives at $\mathbf{x}_d$ from points $\mathbf{x}_L \in \mathcal{L}$ on a light source $\mathcal{L}$ through specular chains $(\mathbf{x}_d, \dots, \mathbf{x}_s, \mathbf{x}_L)$. For readability we write f_s for the product of the BSDF at $\mathbf{x}_d$ and all specular albedos along the chain, and p_σ^B for the product of the BSDF PDF $p_\sigma^{BSDF}(\mathbf{x}_d \to \mathbf{x}_{d+1})$ and all specular selection probabilities. We may leave out the arguments of the incident radiance L_i, the BSDF at $\mathbf{x}_d$, the cosine and p_σ^{BSDF}, if they are evaluated for the direction ω_i sampled from the BSDF, and write $A_B := A(B)$ for the area of a set B.

The BSDF estimator. The variance of the BSDF estimator

$$F_{BSDF} = \frac{L_i f_s \cos}{p_\sigma^B}$$

consists of $\mathbb{E}^2(F_{BSDF}) = L_o^2(\mathbf{x}_d, \omega_o)$ and

$$\mathbb{E}(F_{BSDF}^2) = \int_{\mathcal{L}} \frac{L_e^2(\mathbf{x}_L \to \mathbf{x}_s) f_s^2 \cos}{p_\sigma^B(\omega_i)} \hat{G}(\mathbf{x}_d, \mathbf{x}_L) \, \mathrm{d}A(\mathbf{x}_L), \tag{7.8.1}$$

where $\hat{G}$ consists of the geometry term and Jacobian for transforming from solid angle measure at $\mathbf{x}_d$ to surface area measure at $\mathbf{x}_L$ after refracting and transmitting through the specular chain $(\mathbf{x}_{d+1}, \dots, \mathbf{x}_s)$ as in appendix C.1. We express this as an integration over $\mathcal{L}$ instead of solid angle at $\mathbf{x}_d$ to highlight the influence of the light source size $A_\mathcal{L}$ on the variances.

The regularized estimator. The regularized estimator

$$F_{merge} = \frac{\mathcal{K}_r(\mathbf{x}_L, \mathbf{x}_{nee}) L_e(\mathbf{x}_{nee} \to \mathbf{x}_s)}{p_A^{nee}(\mathbf{x}_{nee})} \frac{f_s \cos}{p_\sigma^B}(\omega_i)$$

only contributes for $\mathbf{x}_L \in \mathcal{L}_\cup$ and $\mathbf{x}_{nee} \in \mathcal{L}_\cap(\mathbf{x}_L)$ with

$$\mathcal{L}_\cup := \bigcup_{\mathbf{x} \in \mathcal{L}} \mathbb{K}_r(\mathbf{x}), \text{ and}$$
$$\mathcal{L}_\cap(\mathbf{x}_L) := \mathcal{L} \cap \mathbb{K}_r(\mathbf{x}_L).$$

The variance of F_{merge} consists of $\mathbb{E}^2(F_{BSDF})$ with

$$\mathbb{E}(F_{merge}) = \int_{\mathcal{L}_\cup} \int_{\mathcal{L}_\cap(\mathbf{x}_L)} \mathcal{K}_r(\mathbf{x}_L, \mathbf{x}_{nee}) L_e(\mathbf{x}_{nee}) \, \mathrm{d}A(\mathbf{x}_{nee}) \cdot f_s(\omega_i) \hat{G}(\mathbf{x}_d, \mathbf{x}_L) \, \mathrm{d}A(\mathbf{x}_L) \tag{7.8.2}$$

and

$$\mathbb{E}(F_{merge}^2) = \int_{\mathcal{L}_\cup} \int_{\mathcal{L}_\cap(\mathbf{x}_L)} \frac{\mathcal{K}_r^2(\mathbf{x}_L, \mathbf{x}_{nee}) L_e^2(\mathbf{x}_{nee})}{p_A^{nee}(\mathbf{x}_{nee})} \, \mathrm{d}A(\mathbf{x}_{nee}) \cdot \frac{f_s^2 \cos}{p_\sigma^B}(\omega_i) \hat{G}(\mathbf{x}_d, \mathbf{x}_L) \, \mathrm{d}A(\mathbf{x}_L). \tag{7.8.3}$$

We now investigate how the two variances behave if we 1) scale the light source while keeping its overall energy constant, i.e. $L_e \propto 1/A_\mathcal{L}$, and 2) scale a uniform kernel with $\mathcal{K}_r = 1/A_{\mathbb{K}_r}$. We assume that both areas are small enough that the Jacobian $\hat{G}$ doesn't affect the variance too much.

The variance of the BSDF estimator is independent of the kernel size. It increases if we shrink the light source while keeping its overall energy constant, since $L_e^2(\mathbf{x}_L)$ contributes quadratically to the integrand in equation 7.8.1 while the integration domain $\mathcal{L}$ only shrinks linearly. In particular, if the light source converges to a point light $\mathcal{L} \to \{\mathbf{x}_p\}$, the variance becomes infinite.

The variance of the regularized estimator depends on both the integrand and the integration domains of equation 7.8.3. The integrand is proportional to $1/A_\mathcal{L}$ and to $1/A_{\mathbb{K}_r}^2$, since $L_e^2 \propto 1/A_\mathcal{L}^2$, $1/p_A^{nee} \propto A_\mathcal{L}$, and $\mathbb{K}_r^2 \propto 1/A_{\mathbb{K}_r}^2$. The integration domains $\mathcal{L}_\cap(\mathbf{x}_L)$ and $\mathcal{L}_\cup$ both depend on both the light and the kernel size. If one is much bigger than the other, the outer domain is mostly determined by the larger area, while the inner domain mostly depends on the smaller area, i.e. $A(\mathcal{L}_\cup) \approx max(A_\mathcal{L}, A_{\mathcal{K}_r})$ and $A(\mathcal{L}_\cap) \approx min(A_\mathcal{L}, A_\mathcal{K})$.

In both cases the inner integrand's dependency on $1/p_A^{nee}$ cancels out with one of the integration domains. As a result the variance is basically independent of the light source size. If the scene contains multiple light sources, the integration domain corresponding to $A_\mathcal{L}$ only partially cancels out with $1/p_A^{nee} = P(\mathcal{L})/A_\mathcal{L}$. We are left with the inverse probability $1/P(\mathcal{L})$ of selecting the light source, which explains why both conventional NEE and our regularized NEE exhibit more variance with additional light sources.

If the kernel size decreases, either $\mathcal{L}_\cup$ or $\mathcal{L}_\cap$ decreases (almost) linearly with it, while $\mathcal{K}_r$ grows quadratically, resulting in $\mathbb{E}(F_{merge}^2)$ and thus the variance growing (almost) linearly with a shrinking kernel area. If the light source converges to a point light $\mathcal{L} \to \{\mathbf{x}_p\}$, the variance converges to the variance of our estimator for two singularities from section 7.5.3.

7.8.2 Area-based Weights

In the previous section we observed that the variance of the BSDF estimator depends (almost) linearly on the light source size, whereas the variance of the regularized estimator depends more or less linearly on the kernel size. We thus propose the *area balance heuristic* and *area power heuristic*

$$w_B^{balance} = \frac{A_{\mathcal{L}}}{A_{\mathcal{L}} + A_{\mathbb{K}_r}} \qquad w_{merge}^{balance} = \frac{A_{\mathbb{K}_r}}{A_{\mathcal{L}} + A_{\mathbb{K}_r}} \tag{7.8.4}$$

$$w_B^{power} = \frac{A_{\mathcal{L}}^2}{A_{\mathcal{L}}^2 + A_{\mathbb{K}_r}^2} \qquad w_{merge}^{power} = \frac{A_{\mathbb{K}_r}^2}{A_{\mathcal{L}}^2 + A_{\mathbb{K}_r}^2}. \tag{7.8.5}$$

Similarity to VCM weights. Extending the area balance heuristic (equation 7.8.4) by $p_A^{nee} = 1/A_{\mathcal{L}}$ results in exactly the same weights as we get from the VCM-style MIS weight derivation (section 7.7). However, our interpretation no longer relies on some factor P_{accept} to be less than 1 in order to represent a meaningful probability.

Accuracy. These weights converge to 0 or 1 if the light source is much bigger or smaller than the kernel, and thus basically disable either BSDF sampling or our merging technique. For similarly large areas ($A_{\mathcal{L}} \approx A_{\mathbb{K}_r}$), our assumption that the integration domains in equation 7.8.3 roughly equal the emitter and kernel area breaks down. Such cases might exhibit lower variance with weights that minimize the variance of the combined estimator instead.

7.8.3 Combined Variance

We aim for MIS weights that solely minimize variance and accept that this may come at the cost of some remaining bias from the regularized technique. In appendix C.3 we present the variance of the combined estimator

$$F_{mis} = w_B(\mathbf{x}_L)\frac{L_i f_s \cos}{p_\sigma^B} + w_m(\mathbf{x}_{L,nee})\frac{\mathcal{K}_r L_i f_s \cos}{p_A^{nee}(\mathbf{x}_{nee}) p_\sigma^B(\mathbf{x}_L)}$$

and show that the variance of F_{mis} is minimized by the BSDF sampling MIS weight

$$w_B \approx \frac{A_{\mathcal{L}_\cap(\mathbf{x}_L)}\frac{(\mathcal{K}_r L_i)^2}{p_A^{nee}}}{A_{\mathcal{L}_\cap(\mathbf{x}_L)}\frac{(\mathcal{K}_r L_i)^2}{p_A^{nee}} + L_i^2} = \frac{A_{\mathcal{L}_\cap(\mathbf{x}_L)}\mathcal{K}_r^2}{A_{\mathcal{L}_\cap(\mathbf{x}_L)}\mathcal{K}_r^2 + p_A^{nee}} =: w_B^{overlap}, \tag{7.8.6}$$

as long as the kernel $\mathcal{K}_r$ is constant and the radiance L_i does not vary too much within $A_{\mathcal{L}_\cap(\mathbf{x}_L)}$. We call this the *overlap heuristic* as it depends on the overlap of the kernel and light source.

Next, we discuss how this weight behaves in the limit for $\mathcal{L} \gg \mathbb{K}$ (e.g. when shrinking the kernel to achieve a consistent estimate) or $\mathcal{L} \ll \mathbb{K}$ (e.g. when modeling point lights) in section 7.8.4. In section 7.8.5 we deal with computing $A_{\mathcal{L}_\cap(\mathbf{x}_L)}$ in various geometric configurations.

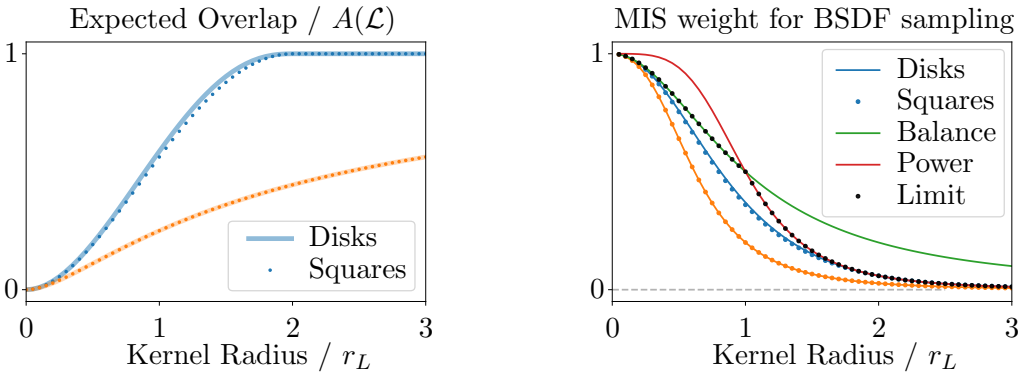


Figure 7.10: **Left**: Expected overlap of two disks (lines) or two squares (dots), where the kernel square's center is inside a square emitter (blue) or anywhere (orange). Overlap and kernel radius are relative to the light source area $A(\mathcal{L})$ and radius r_L, respectively. **Right**: MIS weights for a circular kernel and emitter with the expected disk overlap (blue and orange line), expected square overlap (blue and orange dots), and the balance, power and limit heuristic based on the disk areas.

7.8.4 Limit Analysis

If the kernel is much smaller ($\mathbb{K} \ll \mathcal{L}$) or bigger ($\mathcal{L} \ll \mathbb{K}$) than the light source, then $A_{\mathcal{L}\cap(\mathbf{x}_L)} \approx \min(A_{\mathbb{K}_r}, A_{\mathcal{L}})$. Whenever one region fully covers the other (as is often the case if one is much bigger), this approximation is a true equality. If $\mathbb{K}_r \ll \mathcal{L}$ the weight in equation 7.8.6 converges to the area balance heuristic from equation 7.8.4:

$$w_B^{overlap} \to \frac{\mathcal{K}_r}{\mathcal{K}_r + p_A^{nee}} = \frac{A(\mathcal{L})}{A(\mathcal{L}) + A(\mathbb{K}_r)} = w_B^{balance},$$

and if $\mathcal{L} \ll \mathbb{K}_r$ the weight converges to the area power heuristic from equation 7.8.5:

$$w_B^{overlap} \to \frac{A(\mathcal{L})\mathcal{K}_r^2}{A(\mathcal{L})\mathcal{K}_r^2 + p_A^{nee}} = \frac{A^2(\mathcal{L})}{A^2(\mathcal{L}) + A^2(\mathbb{K}_r)} = w_B^{power}.$$

We thus approximate $A_{\mathcal{L}\cap(\mathbf{x}_L)} \approx \min(A_{\mathcal{L}}, A_{\mathbb{K}_r})$ to form the *area limit heuristic*

$$w_B^{limit} = \begin{cases} w_B^{balance} & \text{if } A(\mathbb{K}_r) \leq A(\mathcal{L}) \\ w_B^{power} & \text{else} \end{cases} \tag{7.8.7}$$

for cases where the overlap computation of $A_{\mathcal{L}\cap(\mathbf{x}_L)}$ is complicated.

7.8.5 Approximating the Overlap

Computing the overlap of simple geometric shapes, such as disks or aligned rectangles, is simple, and we list those formulas in appendix C.4. However, for arbitrary and potentially rotated shapes (e.g. triangles for mesh lights, and ellipses or parallelograms for an anisotropic kernel) this computation is more complicated.

In hopes of a better approximation than the limit cases (equation 7.8.7) we propose to approximate the true overlap in equation 7.8.6 with the average overlap of the two shapes, independent of their translation. Analytically computing this average overlap is often complicated, however we assume that it is similar enough to the average overlap

$\mathbb{E}(A_o^{squares})$ of two aligned squares with corresponding areas. We support this assumption with empirical observations in figure 7.10 comparing the overlap of two disks to that of two spheres, and in section C.5.1 showing several brute-force comparisons for less well-behaved shapes. The expected overlap of two squares with radii r_l, r_k is

$$\mathbb{E}_{\mathbf{x}_L \in \mathcal{L}_\cup}(A_o^{squares}) = 4\frac{r_l^2 r_k^2}{(r_l + r_k)^2}.$$

If the center of the kernel is inside the light, the expected overlap is

$$\mathbb{E}_{\mathbf{x}_L \in \mathcal{L}}(A_o^{squares}) = \begin{cases} 4r_l^2 & \text{if } r_k > 2r_l \\ \left(2r_l r_k - 0.5r_k^2\right)^2 / r_l^2 & \text{else.} \end{cases}$$

We consider this second case since technically the variance for $\mathbf{x}_L \notin \mathcal{L}$ is minimal for $w_B = 1$ where the contribution of F_{BSDF} is 0.

We approximate $A_{\mathcal{L}_\cap(\mathbf{x}_L)} \approx \mathbb{E}(A_o^{squares})$ to form the *average square overlap heuristic*,

$$w_B^{avg} = \frac{\mathbb{E}(A_o^{squares})\mathcal{K}_r^2}{\mathbb{E}(A_o^{squares})\mathcal{K}_r^2 + p_A^{nee}}, \tag{7.8.8}$$

with either $\mathbf{x}_L \in \mathcal{L}$ or $\mathbf{x}_L \in \mathcal{L}_\cup$, and we compute the radii r_l, r_k of the two squares as $r_{l,k} = \sqrt{A_{\mathcal{L},\mathbb{K}_r}}/2$ from the emitter and kernel areas $A_\mathcal{L}, A_{\mathbb{K}_r}$. Figure 7.10 (right) indicates that this provides a rather close fit to the true overlap heuristic from equation 7.8.6 for well-behaved scenarios, and section C.5.1 shows that it is otherwise at least a closer approximation than the area limit heuristic that simply uses the kernel or emitter areas.

7.8.6 Forcing the BSDF Hemisphere

We can replace the merge strategy with an upper- and lower-hemisphere merge at the last specular vertex as in section 7.5.4. In this case p_σ^B no longer fully cancels out in equation 7.8.6 and we get

$$w_B = \frac{A_{\mathcal{L}_\cap(\mathbf{x}_L)}\mathcal{K}_r^2}{A_{\mathcal{L}_\cap(\mathbf{x}_L)}\mathcal{K}_r^2 + p_A^{nee}/\mathbb{P}^{+,-}},$$

with either $\mathbb{P}^+$ or $\mathbb{P}^-$, depending on whether $\mathbf{x}_L$ is in the upper or lower hemisphere of $\mathbf{x}_s$. If $\mathbb{P}^+$ is the Fresnel term, this results in

$$w_B = \frac{F \cdot A_{\mathcal{L}_\cap(\mathbf{x}_L)}\mathcal{K}_r^2}{F \cdot A_{\mathcal{L}_\cap(\mathbf{x}_L)}\mathcal{K}_r^2 + p_A^{nee}}$$

for upper-hemisphere samples, and replacing F with $(1-F)$ for lower-hemisphere samples. For the area limit heuristic and average overlap heuristic we only need to replace $A_{\mathcal{L}_\cap(\mathbf{x}_L)}$ with the corresponding overlap approximation.

7.8.7 Multiple Emitters

If there are multiple emitters, the inner integration domain $\mathcal{L}_\cap(\mathbf{x}_L)$ now refers to the overlap of the kernel with all the light sources. This might involve placing the kernel

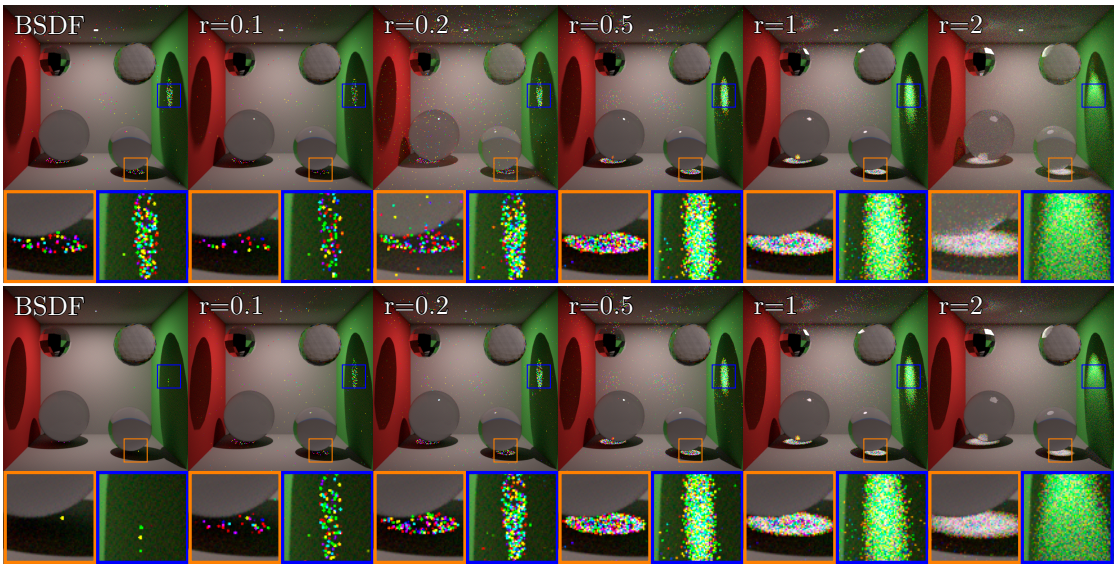


Figure 7.11: Disk kernel with different fixed radii at 100 spp, and a medium small (**top**) and very small (**bottom**) light source. With a medium small light source, a small regularization radius performs worse than BSDF sampling.

at multiple centers along a specular ray and rotating it to match nearby light sources. In addition p_A^{nee} is not necessarily constant anymore, and does not translate to the area heuristics from section 7.8.4. The latter is fairly easy to fix by using p_A^{nee} directly instead of $1/A_{\mathcal{L}}$ in the area heuristic weights. However, weights based on the true kernel-light-overlap (equation 7.8.6) or on an expected average overlap (equation 7.8.8) become more complicated and could be a potential topic for future work.

7.9 Results

All results were rendered using our own spectral renderer, and we only evaluate our regularization technique from specular surfaces to area light sources and environment maps. We investigate the influence of the regularization kernel scale (section 7.9.1) and shape (section 7.9.2) in isolation, biased visibility (section 7.9.3) and issues with multiple light sources (section 7.9.4), before briefly discussing environment maps (section 7.9.5) and then delving into the different MIS weighting strategies (section 7.9.6) and the pros and cons of a consistent regularization strategy (section 7.9.7).

7.9.1 Kernel Scales

Figure 7.11 shows how our regularization technique (without MIS) affects bias. If the radius is too large we get smooth images but strong blurring and larger, less sharp highlights and caustics. If the radius is too small we may get higher variance than non-regularized BSDF sampling if the light source is big enough (top left).

If a caustic is caused by a finite specular surface – such as the caustics on the ceiling in figure 7.12 – increasing the kernel size beyond a certain scale no longer reduces noise. This is because our technique only helps with paths that hit a specular facet and miss the light souce by some margin, and not with paths that miss the specular facet in the first

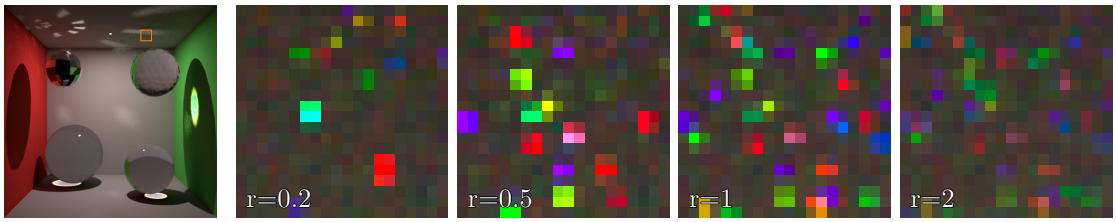


Figure 7.12: Constant disk kernel with radius $0.2, 0.5, 1$ and 2. We can observe a variance reduction from $r = 0.2$ to $r = 1$ due to additional accepted connections. However, since we can only regularize paths that hit the appropriate facet in the top left mirror sphere, and a disk kernel with radius 1 is already big enough to accept all of these paths, radius 2 produces no additional paths and only decreases the caustic's intensity due to weighting with a smaller kernel value.

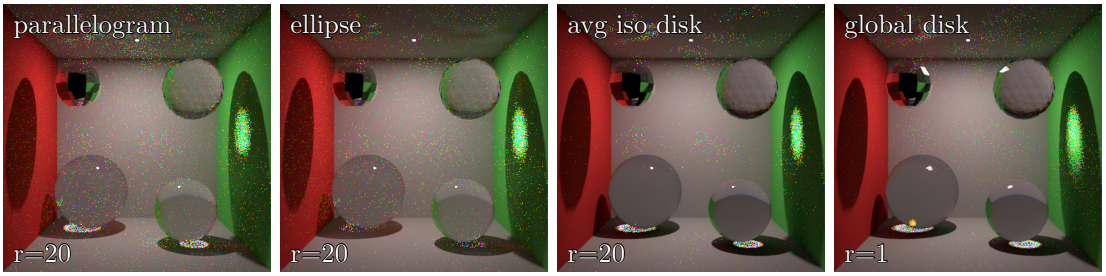


Figure 7.13: Kernel shapes. **Left to right**: Parallelogram and ellipse based on anisotropic footprint, disk based on average footprint, disk with global radius.

place. In terms of the variance analysis in equations 7.8.2 and 7.8.3, such finite surfaces restrict the set $\mathcal{L}_\cup$ to the subset of the kernel-extended light source that is visible through the facet. This area is limited by the facet, so it stops growing once the kernel exceeds a certain scale, and we do not get any additional accepted connections. From this point on the variance only decreases further because samples are weighted down with a smaller kernel value.

7.9.2 Kernel Shapes

Figure 7.13 was rendered with different kernel shapes for our technique. Figure 7.14 compares our kernel to roughness regularization.

Comparison to cone mollification. Figure 7.15 compares our technique to the regularization technique proposed by [20]. The size and sharpness of the caustics and highlights are similar. The main difference is the caustic shape, since we align our kernel with the light source plane, whereas [20] align their kernel perpendicular to the specular direction, resembling a scene with a disk-shaped light source oriented in that direction.

Constant disk vs. adaptive parallelogram. Figure 7.16 compares a constant disk kernel with an adaptive parallelogram kernel constructed from the scaled ray differentials (see section 7.6.1 and figure 7.8). The disk radius and parallelogram scale were hand-picked such that the caustic on the top left has similar sharpness for both.

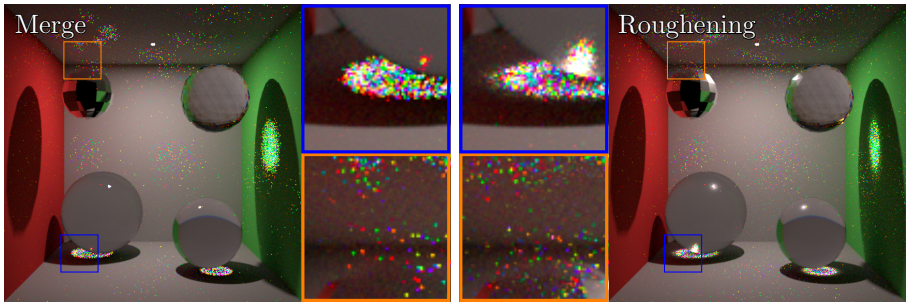


Figure 7.14: Our kernel (parallelogram based on anisotropic footprint, **left**) and roughness regularization (**right**). Both techniques only introduce bias where necessary. With our techniques caustics grow in size and get weaker overall while staying fairly sharp. Roughness regularization blurs the caustics and highlights.

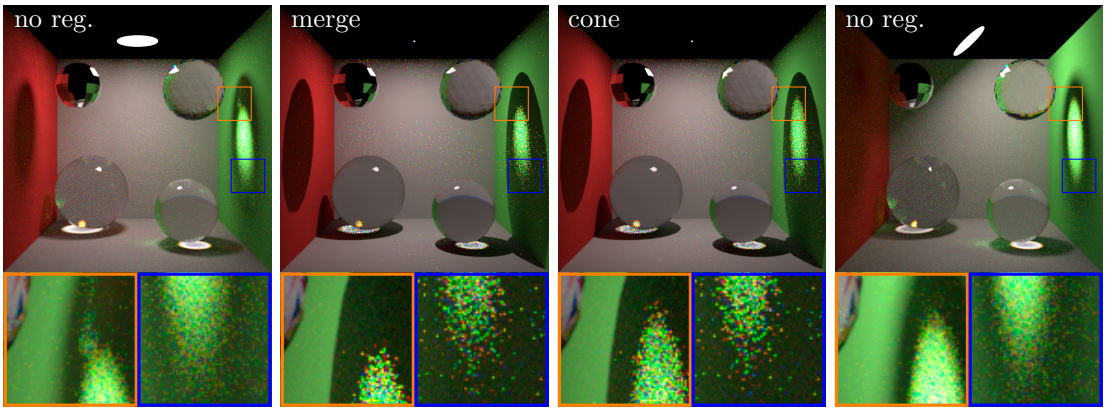


Figure 7.15: Outer images: No regularization, 1024 spp. Inner images: Our regularization technique (**left**) and [20] (**right**), 128 spp. The kernel size for the inner images equals the disk light size in the outer images without regularization. The caustics are similarly sized and sharp, but have different shapes.

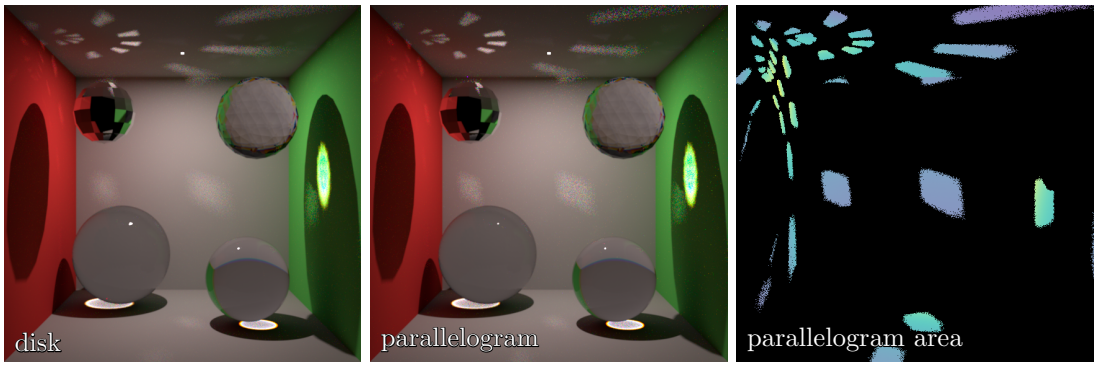


Figure 7.16: **Left**: Disk kernel with constant radius = 0.2. **Center**: Parallelogram kernel scaled by 5. **Right**: Area of the parallelogram kernel for direct caustics from the upper left sphere. While the caustics to the top left have similar sharpness, the caustics on the top right and back wall appear more blurred with the constant disk kernel.

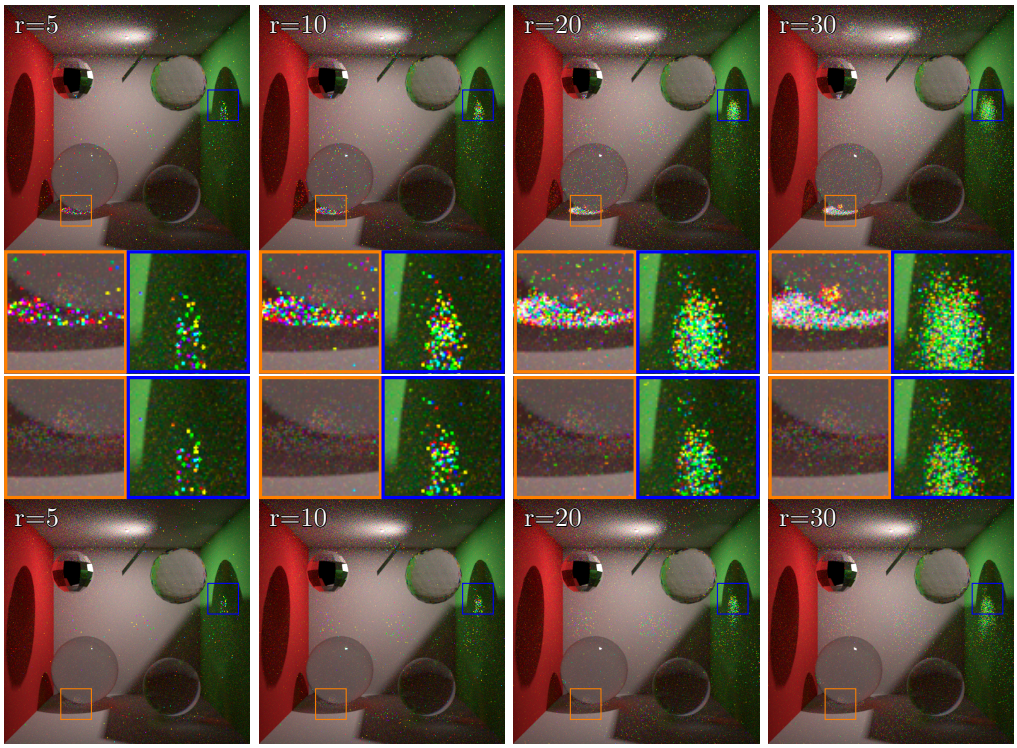


Figure 7.17: Visibility evaluation for the NEE vertex (**top**) and BSDF direction (**bottom**). Parallelogram kernel scaled by $r = 5, 10, 20, 30$, 100spp, no MIS.

7.9.3 Visibility

In figure 7.17 we evaluate visibility for the NEE vertex and for the specular direction. Either option may result in light leaking: Testing visibility for the NEE vertex might cause caustics from blocked specular BSDF directions, whereas testing visibility for the BSDF direction might cause caustics from occluded emitters. In the limit both converge to the same result, and we test for visibility with the NEE vertex in the remaining results.

7.9.4 Multiple Light Sources

Adding more emitters to a scene may increase the variance of our estimator combined with naive NEE sampling. The following examples show when this might be the case, depending on the emitter's energy and relative influence.

Even illumination. The scene in figure 7.18 shows the mirror image of a grid of emitters and of a colored plane that is illuminated through the mirror. The emitter energy is scaled to maintain overall brightness in the images. In this case the variance on the plane stays unaffected, since all emitters have a similar influence. Thus the overall probability for a successful regularization stays roughly the same, since the lower acceptance probability for any particular light is offset by the equally growing set of directions where any regularization is possible. For any particular light, the variance increase due to fewer successful regularizations to this light is canceled out by the variance decrease due to the lower emission and thus lower overall path contribution.

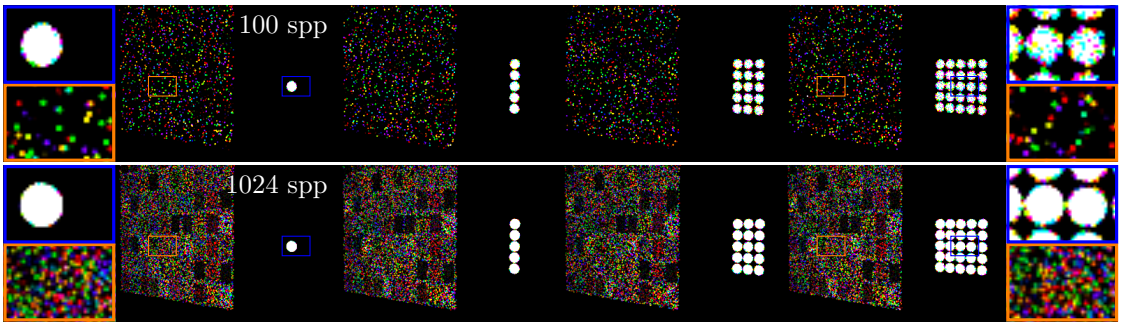


Figure 7.18: A diffuse surface and grid of emitters reflected by a mirror. On the surface on the left (orange inset) all light sources have similar influence, so using more of them (with lower emission energy per light source to maintain overall brightness) hardly changes the variance. Each direct reflection of a light source (blue inset) only depends on one light source, so adding more lights also adds more variance (blue insets in top row). Elliptical kernel based on anisotropic footprint, scaled by $r = 10$, no MIS.

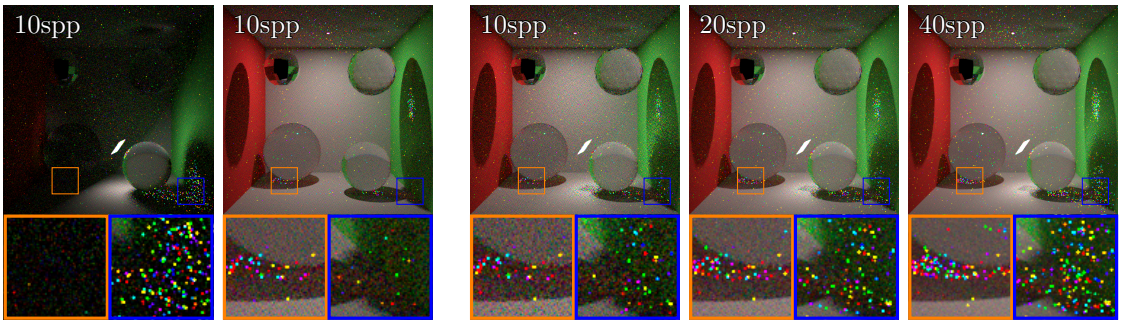
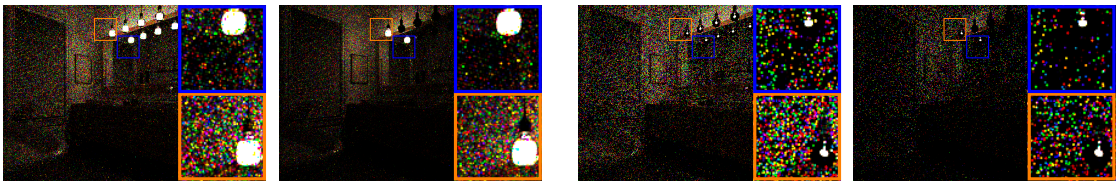


Figure 7.19: **Left**: Individual emitters with different overall energy. **Right**: Both emitters at equal, double and quadruple sample count. Since the emitters affect different regions, having both in the same scene increases the variance. Elliptical kernel based on anisotropic footprint, scaled by $r = 10$, no MIS.

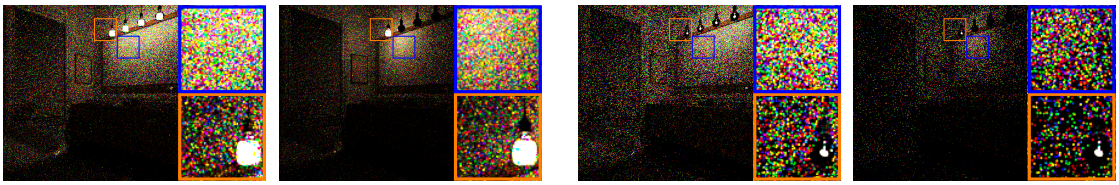
Uneven illumination. Variance increases in regions with varying emitter influence, in particular if regions are affected strongly by emitters with overall lower energy, i.e. lower p_A^{nee}. For example, when rendering the caustic in the blue inset in figure 7.19 we only get a fraction of the successful regularizations for caustic paths when rendering the scene with two light sources, since many of the NEE samples for potentially acceptable caustic directions are on the brighter top light source and contribute nothing. Ultimately, this problem is orthogonal to our method and can appear in any scene where weak lights have a strong influence on some parts of the scene, but are unlikely to be selected from energy-based NEE sampling.

The bathroom scene. Figures 7.20 and 7.21 contain 4 equally-sized area lights with slightly varying emission encased in glass bulbs. In this scene the naive NEE sampling slightly increases the variance of our technique with 4 light sources instead of one, though not as badly as in figure 7.19 since more distant light sources still have some impact on the wall in the insets.



Anisotropic kernel, r=15 No regularization

Figure 7.20: Bathroom scene with 1024 spp. The regularized technique (anisotropic parallelogram kernel scaled by 15, no MIS) exhibits lower variance with only one light source since regularization never fails due to sampling the wrong light source.



Anisotropic kernel, r=15 No regularization

Figure 7.21: Scene from figure 7.20 without a mirror, 1024 spp.

7.9.5 Environment Maps

The pool scene in figures 7.22, 7.23, and 7.24 is illuminated by an environment map. In these renders we prefilter the environment map for a kernel in texture space. Figure 7.22 shows results for varying kernel scales for an elliptical kernel based on the directional ray differentials. Figure 7.23 compares the results for the same anisotropic kernel with an isotropic disk kernel based on averaging the directional ray differentials. Figure 7.24 shows the anisotropic kernel at different scales and water levels.

7.9.6 MIS

Figure 7.25 compares some of the MIS techniques that we discussed in sections 7.7 and 7.8. Section C.5.2 shows all of them. We start with a light source that is just big enough to work reasonably well without regularization (top), scale it by 0.5 where the merged and unregularized samples have similar variance (center), and scale it by 0.5 again (bottom) which makes it just small enough that we want the MIS weights to favor regularized samples.

Unfortunately providing RMSE plots for all techniques is impractical, since all MIS strategies converge to results with different bias. This means we would have to compute a reference for each individual strategy, kernel size and light source size, many of which produce rather noisy images and would require a long time to converge.

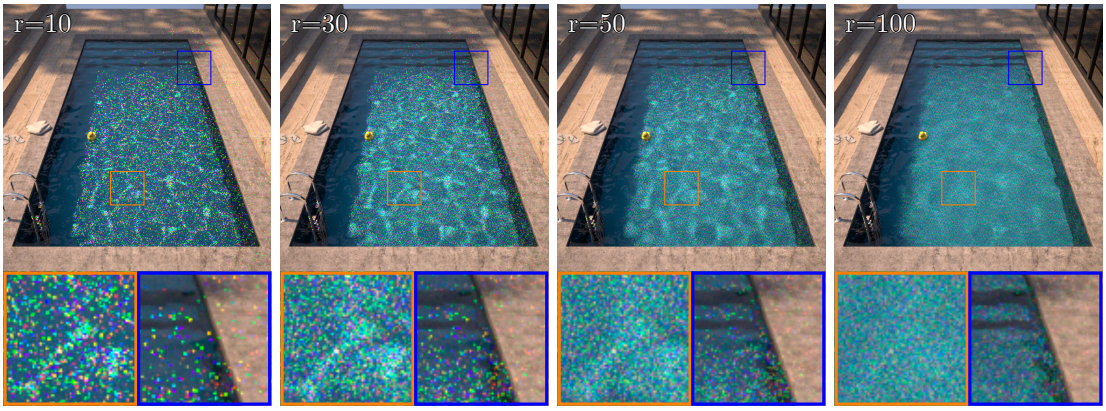


Figure 7.22: The pool scene with anisotropic ellipse kernel prefiltering in texture space. As the kernel scale grows (**left to right**), noise decreases but the caustics become more blurry.

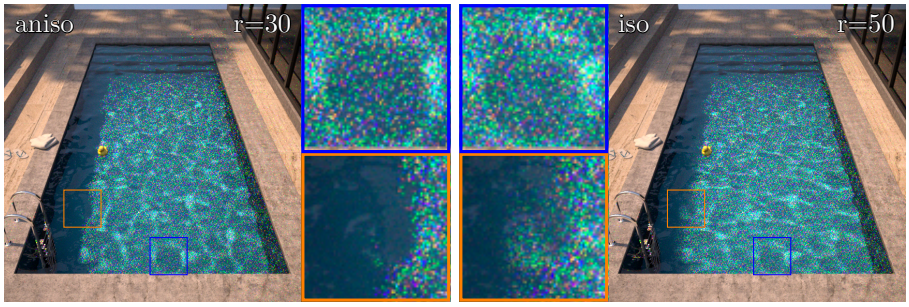


Figure 7.23: The pool scene with different kernel shapes. Note how the caustic jumps between the anisotropic and isotropic prefiltering. As the kernels have different areas at equal scales, we show similar areas instead of scales for better comparison.

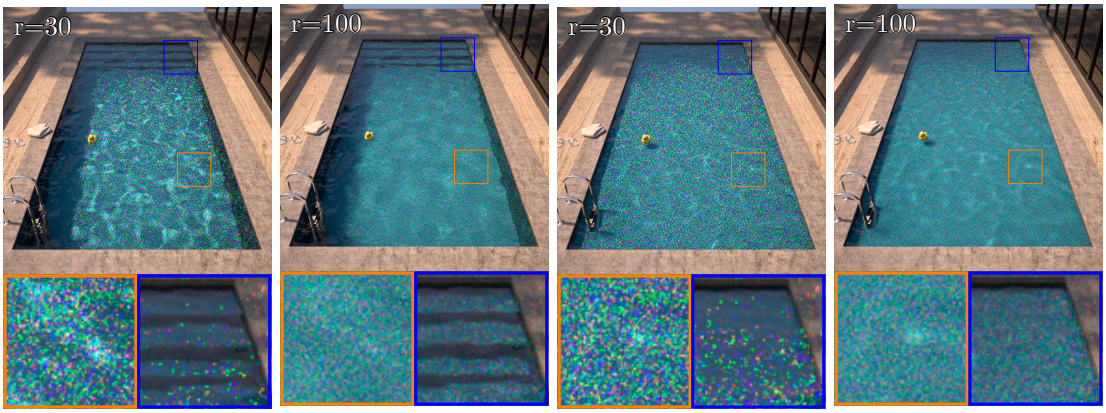


Figure 7.24: The pool scene with different floor levels and kernel scales. If the water is deep (**left**), a smaller kernel scale helps preserve the sharp caustics. With shallow water (**right**), the caustics are weak and the larger kernel helps blurring out noise without loosing much detail.

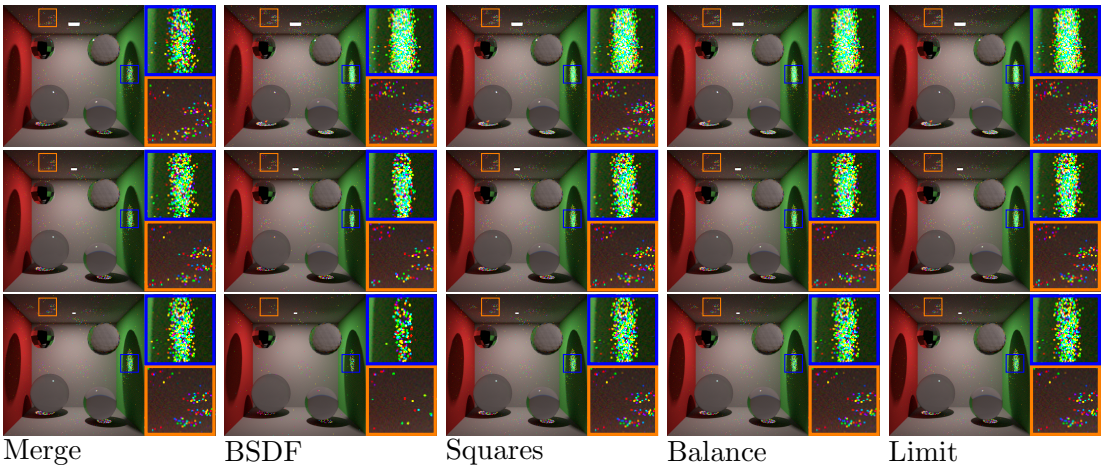


Figure 7.25: **Left to right**: Merge only, BSDF sampling only, three MIS strategies. The light source on top is scaled by 1/2 (**center**) and 1/4 (**bottom**). Section C.5.2 shows more comparisons.

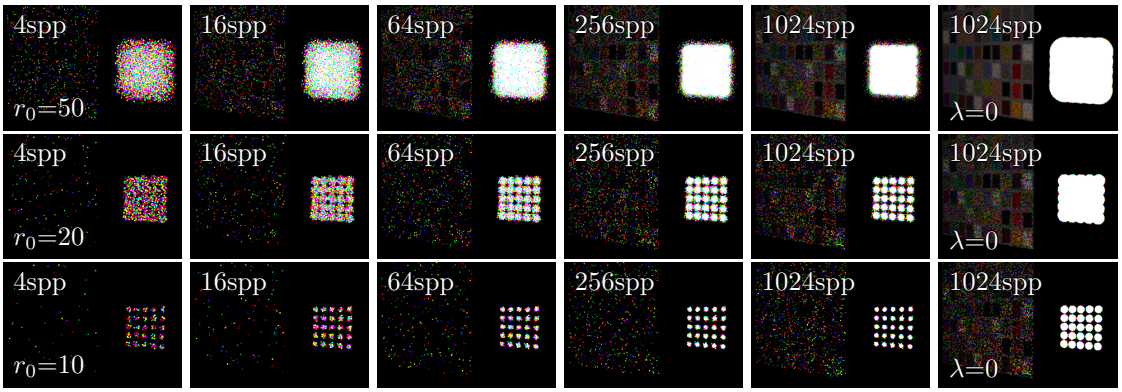


Figure 7.26: Ellipse kernel, initial scales 50 (**top**), 20 (**center**) and 10 (**bottom**), shrinking rate $\lambda = -1/6$, no MIS. The last column uses the initial scale for all iterations without shrinking ($\lambda = 0$).

7.9.7 Consistency

All previous results were rendered with a constant kernel scaling to better illustrate the influence of different kernel and light source sizes. Our technique can be made consistent by shrinking the kernel over iterations, as illustrated by figures 7.26 and 7.27, where we shrink $r_n = n^\lambda r_0$ at a rate of $\lambda = -1/6$ as suggested by [20] and [36]. However, this also means that strong bias from an initially large kernel stays in the image for a long time, accompanied by high variance once the kernel shrinks below a size that offers a good trade-off between bias and variance.

In figure 7.26, a constant scale of $r_n \equiv 10$ (bottom right) produces a smoother reflection than any of the shrinking kernels, and sharper highlights than renders starting with a larger kernel that still have leftover bias from early iterations. Figure 7.27 also shows how our method behaves with MIS (equation 7.8.8). We get faster initial variance reduction than BSDF sampling, but with MIS we never fall behind BSDF sampling in the long term when our regularized technique fails more often due to the shrinking kernel.

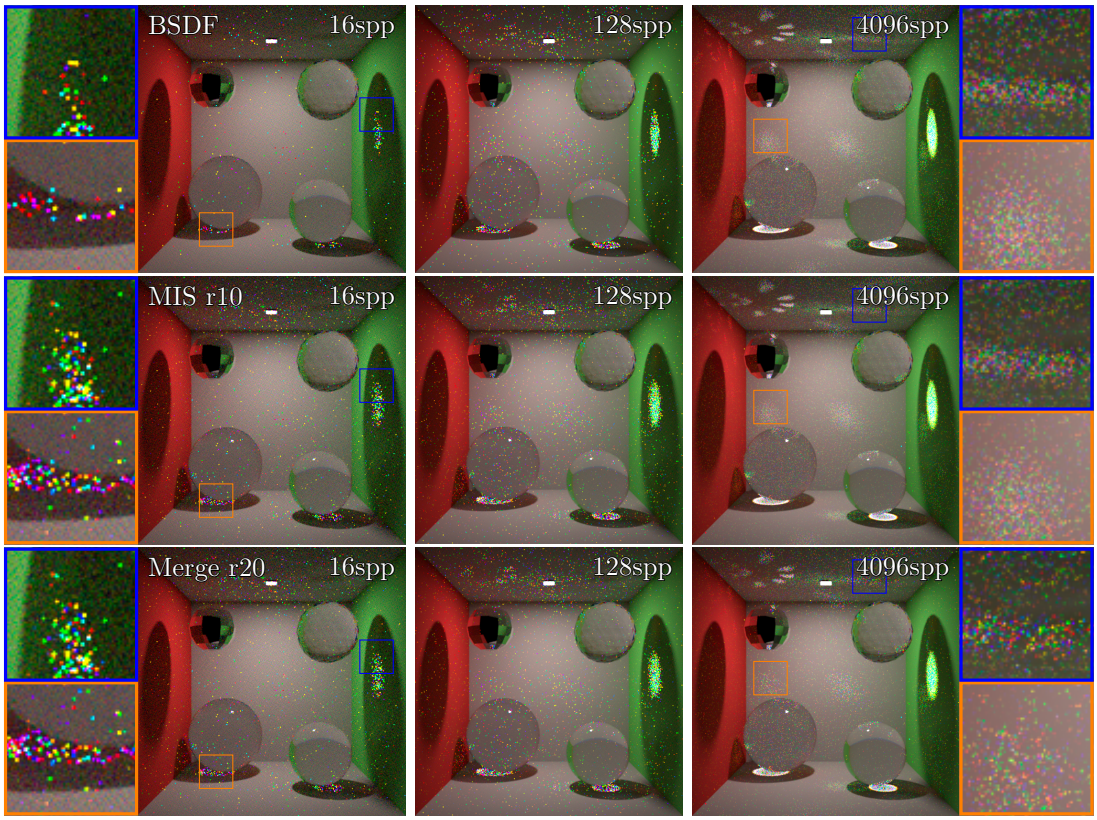


Figure 7.27: Parallelogram kernel, initial scale $r_0 = 10$, shrinking rate $\lambda = -1/6$. With MIS (**center**), we benefit from the initial variance reduction of our merge (**bottom left**, note the caustics), and still keep up with BSDF sampling (**top**) in the long run (note e.g. the right half of the ceiling) when the regularization fails more often because of the shrinking kernel (**bottom right**).

7.10 Limitations and Future Work

Our theoretical framework can regularize any kind of NEE connection, but we only derived and implemented MIS weights for connections from specular to area lights. We did not explore MIS with connections from glossy surfaces or to environment lights. Applying our MIS derivation to glossy surfaces should be straightforward, but applying it to environment maps might prove challenging since computing the solid angle of our texture space kernels is not trivial.

Overlapping emitters. In scenes with multiple light sources p_A^{nee} typically depends on the light source's area and power. In principle we can use this in the acceptance probability (equation 7.7.1) and use the VCM-style MIS weights (equation 7.7.2), although the approximation of the acceptance probability becomes less accurate if multiple light sources within regularization distance occlude each other (see figure 7.9). Some sort of cone tracing [140], with a cone opening angle corresponding to the kernel scale, might help identify light sources that can potentially be regularized, and a closer approximation to the true acceptance probability might improve the results.

Many lights. Conventional NEE struggles in scenes with glossy surfaces and many and/or large light sources, since the NEE vertex rarely matches the peak of the glossy BSDF. For the same reason our technique struggles with many and/or large light sources, where we are less likely to sample an NEE vertex near a specular direction.

If the light sources are large, good paths can be sampled easily with BSDF sampling, and the variance of our technique is weighted down through MIS. If the light sources are small, BSDF sampling will usually miss them as well, but at least our technique gives us some additional chance of finding such paths. Even with glossy surfaces where conventional NEE is possible, such NEE connections will often be off the glossy peaks, and using our technique to merge the NEE sample with a glossy BSDF direction might be helpful.

In either case, some sort of light source guiding or cone tracing might help sample NEE samples with a high chance of successful regularization. With guided sampling we would simply need to replace p_A^{nee} with the guided PDF in all the equations.

7.11 Conclusion

We presented a new regularization scheme for specular next event estimation between specular surfaces and area lights, together with a framework to regularize any type of surface-to-emitter connection. Our derivation follows the ideas of VCM but shifts the merge from the last camera path vertex to the last (and only) light vertex, and allows for either vertex to be specular.

For point lights our scheme is equivalent to the BSDF cone regularization proposed by Kaplanyan et al. [20]. However, since we align our regularization kernel with the light source, our technique naturally extends to area lights and allows for MIS between our technique and unregularized strategies.

We derived MIS weights to minimize the variance of the combined estimator, and showed that they mainly depend on the relative size of the light source area and kernel area. The weights turned out to be impractical to compute but can easily be approximated using the average overlap between simple geometric shapes. In practice, our weights perform very similar to the VCM balance heuristic.

Shrinking the regularization kernel makes the technique consistent but may end up with a worse performance than a well-picked constant kernel scale. In this case using MIS at least ensures that our technique never falls behind unregularized BSDF sampling.

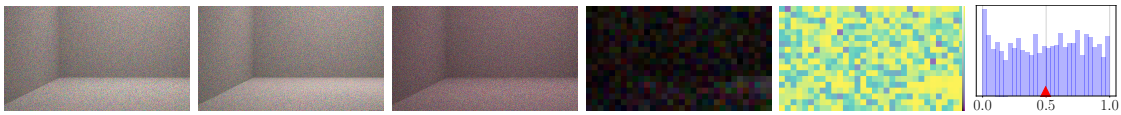
Just like normal NEE, our method struggles with multiple light sources with different contributions to different parts of the scene. We believe that our method should integrate well with some sort of guided light source or irradiance sampling that samples NEE vertices in the rough proximity of (near) specular rays. Then our merge or some other regularization technique can give the final nudge for NEE samples that are close to but not quite in the specular direction.

8 Detecting Bias in Monte Carlo Renderers using Welch's t-test

This chapter is based on
A. Jung, J. Hanika, and C. Dachsbacher (2020) "Detecting Bias in Monte Carlo Renderers using Welch's t-test", In: *Journal of Computer Graphics Techniques (JCGT)* 9.2

When developing a new renderer we usually want a way to check if it was implemented correctly. Conventionally this is done by comparing to the output of a reference implementation. However, such tests require a large number of samples to be reliable, and sometimes are unable to reveal very subtle differences that are caused by bias but overshadowed by random noise.

We propose using a statistical test, Welch's t-test [7], to reliably find small bias even at low sample counts. Welch's t-test is an established method in statistics to determine if two sample sets have the same underlying mean, based on sample statistics. We adapt it to test whether two renderers converge to the same image, i.e. the same mean per pixel or pixel region. We also present two strategies for visualizing and analyzing the test's results, assisting us in localizing especially problematic image regions and detecting biased implementations with high confidence at low sample counts both for the reference and tested implementation.



(a) PT (b) BDPT (c) Diff ×8 (d) Block diff (e) Color map (f) p-values

Figure 8.1: Welch's t-test in an empty, gray box, testing a biased bidirectional path tracer (b) against an unbiased forward path tracer reference implementation (a), both limited to paths with 3 vertices, with 10 samples per pixel each. The two center images show the difference (c) and tile-wise difference (d), which hardly reveal any bias. Welch's t-test outputs a color map (e) revealing bias to the right, and a non-uniform histogram of p-values (f) as strong evidence that (a) and (b) will not converge to the same image with more samples.

8.1 Introduction

The underlying integrals for generating photorealistic imagery are only solvable using statistical methods, the most common being Monte Carlo integration. In this chapter, we focus on verifying that an implementation of a Monte Carlo estimator does what it is supposed to do. We only consider unbiased estimators, i.e. estimators whose expected value at any final sample count is equal to the true solution of the integral that needs to be solved – at least, if the estimator was designed and implemented correctly.

Conventional tests for verifying the correctness of a new renderer implementation (see section 4.9.3) are typically based on comparisons to a reference implementation. However, strong noise at lower sample counts can make those tests unreliable at low computation times. Even at longer computation times, subtle bias may still be hidden behind weak remaining noise.

We propose using a statistical test, Welch's t-test, to test whether an implementation of an unbiased renderer does indeed produce an unbiased result when tested against an unbiased reference implementation. The test's results help detect bias and therefore faulty implementations by considering statistics of the samples generated during rendering. Welch's t-test is a two-sided hypothesis test to decide whether the underlying means of two normal-distributed sample sets are equal. In the context of rendering, a correctly implemented unbiased Monte Carlo estimator converges to the underlying mean of the samples drawn during rendering. We can thus use Welch's t-test to test whether the underlying mean of the distribution sampled by the Monte Carlo estimator is equal to the mean of a reference implementation. The test may only be used to check whether a new implementation is correct, while for comparing convergence speed one still has to resort to other methods such as root mean squared error (RMSE, equation 4.9.1) over time or sample count.

Our contributions are:

- A detailed description of the steps necessary for applying Welch's t-test to test implementations of supposedly unbiased renderers for bias.
- A method for converting samples generated by a Monte Carlo renderer to samples that can be used as input to Welch's t-test.
- Two visualization schemes for analyzing the test's result.
- An evaluation of the test's behavior under various settings.

8.2 Related Work

Typical quantitative metrics for comparing two images are the root mean square error (*RMSE*, see section 4.9.3) or the absolute difference image, both of which are prone to initial noise hiding minor bias and may require large sample counts for a reliable evaluation. In the context of predictive rendering renderers are validated with respect to measured data, whereas we validate correctness with respect to the output of a reference implementation.

Statistical methods. To our knowledge, Subr and Arvo [141] are the only ones to employ classical statistical methods for testing renderers, aside from the MSE estimation done by [113]. They propose using various statistical tests to analyze variances and means of renderers, including Welch's t-test to test for equal means. Their work covers the basics of statistical hypothesis testing and also tests isolated components such as the BSDF, but Welch's t-test and its application are only briefly discussed. We provide a more in-depth explanation of the test and steps necessary for applying it to Monte Carlo renderers, as well as a more extensive analysis including visualization of the test results.

8.3 Welch's t-Test

Given two sets $\{X_{1,1}, \ldots, X_{1,N_1}\}$, $\{X_{2,1}, \ldots, X_{2,N_2}\}$ with N_1 and N_2 individual independent random samples, where each sample set was drawn from a normal distribution with unknown individual means μ_1, μ_2 and variances σ_1, σ_2, Welch's t-test can be used to test either of the hypotheses $\mu_1 = \mu_2$ or $\mu_1 \geq \mu_2$. Unlike similar tests, it does not require the variances σ_1, σ_2 of the underlying distributions to be equal.

We use the two-tailed Welch's t-test for testing if $\mu_1 = \mu_2$. First, we compute a statistic of the sample sets. If $\mu_1 = \mu_2$, this statistic is distributed according to a known two-tailed distribution. This allows us to evaluate the probability of observing the measured sample sets under the tested hypothesis $\mu_1 = \mu_2$, which then lets us draw conclusions about the hypothesis' plausibility.

Welch's t-test is based on the sample sets' *sample means* $\bar{X}_1, \bar{X}_2$ (equation 4.2.7) and *unbiased sample variances* S_1^2, S_2^2 (equation 4.2.8). They get combined into the t-statistic

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{S_1^2}{N_1} + \frac{S_2^2}{N_2}}}. \tag{8.3.1}$$

The t-statistic is itself a random variable and, if $\mu_1 = \mu_2$, it is distributed according to the t-distribution

$$f_\nu(t) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\,\Gamma\left(\frac{\nu}{2}\right)} \left(1 + \frac{t^2}{\nu}\right)^{-\frac{\nu+1}{2}} \tag{8.3.2}$$

with the appropriate *degrees of freedom* $\nu \in \mathbb{N}$. Γ is the Gamma function, a generalization of the factorial. The degrees of freedom ν resulting in the t-distribution of the measured t-statistic can be approximated by the Welch-Satterthwaite equation [142], based on the measured sample variances and sizes, as

$$\nu = \frac{\left(\frac{S_1^2}{N_1} + \frac{S_2^2}{N_2}\right)^2}{\frac{S_1^4}{N_1^2 \cdot (N_1 - 1)} + \frac{S_2^4}{N_2^2 \cdot (N_2 - 1)}}. \tag{8.3.3}$$

Figures 8.2 and 8.3 show the t-distribution for increasing values of ν.

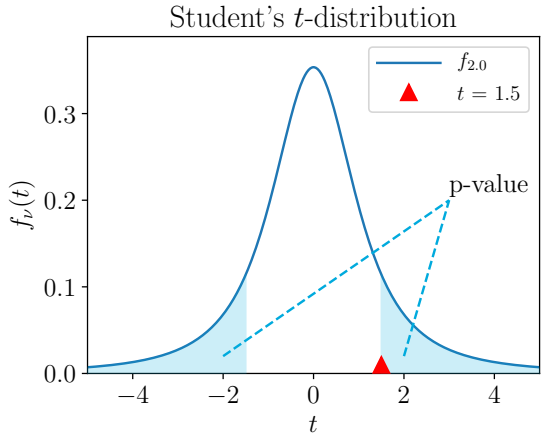


Figure 8.2: Two-tailed Welch's t-test where $\nu = 2$ and the t-statistic of the observed sample sets is 1.5. The p-value is the area of both tails of the t-distribution beyond t, which is ≈ 0.27.

8.3.1 Testing for Equal Means

Now we can test whether the means of the underlying distributions (not the sample means) are equal. We call this the *null hypothesis*

$$H_0 : \mu_1 = \mu_2$$

and test it against the *two-sided alternative hypothesis*

$$H_a : \mu_1 \neq \mu_2.$$

"Two-sided" simply means that if $\mu_1 \neq \mu_2$ we do not care which one is bigger.

The essential insight of Welch's t-test is that if the null hypothesis H_0 is true, the t-statistic computed from the observed sample sets (equation 8.3.1) approximately follows the t-distribution (equation 8.3.2) with degrees of freedom approximated using equation 8.3.3. We can use this knowledge to compute the probability $p_\nu(t)$, called the p-value, of measuring the observed t-statistic or something more extreme (i.e. any t' with $|t'| \geq |t|$) for the measured t-statistic t computed from the available samples:

$$p_\nu(t) = 2 \cdot \int_{|t|}^{\infty} f_\nu(\bar{t}) d\bar{t}. \tag{8.3.4}$$

A p-value below a certain threshold (e.g. $a = 0.01$) suggests it is rather unlikely that two sample sets, drawn from normal distributions with equal means, result in the t-statistic computed from the observed sample sets, and as such, it is unlikely that the two sample sets were indeed drawn from normal distributions with equal means. In turn, this indicates H_0 is likely false and we can reject it with a confidence of $(1 - a) \cdot 100\%$. In other words, the probability of falsely rejecting H_0 is $a \cdot 100\%$.

Visually speaking, the p-value equals two times the area of the tail of the t-distribution beyond the computed t-statistic (figure 8.2).

In summary, in order to conduct Welch's t-test, we need to:

1. Measure two normal-distributed sample sets,
2. compute their respective sample means and sample variances,
3. compute the t-statistic,
4. compute the degrees of freedom ν, and
5. compute the p-value based on t and ν

For evaluating the integral in equation 8.3.4 we use the algorithm presented by Cooper [143]. If conducting a single test, a p-value below a user-defined threshold can then be used to reject H_0. However, note that Welch's t-test can only be used to detect false hypothesis; a p-value above some threshold does not proof H_0 to be true.

In the context of rendering, we will conduct separate tests per image region and color channel, and combine their results to gain more conclusive insights in section 8.4.

8.3.2 Properties of the p-Value

Since the p-value depends on t and ν, which in turn depend on the samples, the p- value is a random variable itself. As p describes the probability of observing the measured t or something more extreme given H_0, if H_0 is indeed true, p should take any value in $[0, 1]$ with equal probability. It turns out that if the null hypothesis is true (i.e. the two distributions have the same underlying mean), the p-value is actually uniformly distributed. This also implies that p is smaller than some $a \in [0, 1]$ with a probability of $a \cdot 100\%$, or $P(p < a) = a$.

Therefore we can reject H_0 if $p < a$ for some threshold a with confidence $(1-a)\cdot 100\%$: The probability of falsely rejecting a true null hypothesis due to $p > a$ is $a \cdot 100\%$, as the probability of measuring a $p < a$ is only $\int_0^a 1 \, \mathrm{d}x = a$.

We will use this criterion indirectly, not by conducting multiple tests with different samples, but by comparing the p-values resulting from applying Welch's t-test to multiple image regions. Each image region corresponds to a different distribution. If the estimator (i.e. the renderer) is unbiased in each region, i.e. both the reference and the tested estimator have the same expected value, each region's p-value is uniformly distributed, and therefore the p-values taken from the different image regions are uniformly distributed as well. This allows us to detect a biased estimator through a non-uniform distribution of p-values taken from different image regions.

This line of reasoning does not work the other way around: A uniform distribution of p-values taken from different image regions does not imply that the individual p-values are uniformly distributed for their respective image regions. So while non-uniformly distributed p-values from different image regions imply bias, uniformly distributed p-values are no guarantee that the estimators are unbiased.

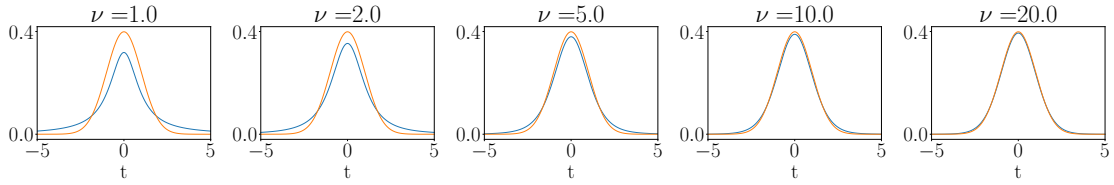


Figure 8.3: The t-distribution for $\nu = 1, 2, 5, 10, 20$ (blue) and the standard normal distribution f_∞ ($\mu = 0, \sigma = 1$) (orange). As $\nu \to \infty$ the t-distribution approaches f_∞.

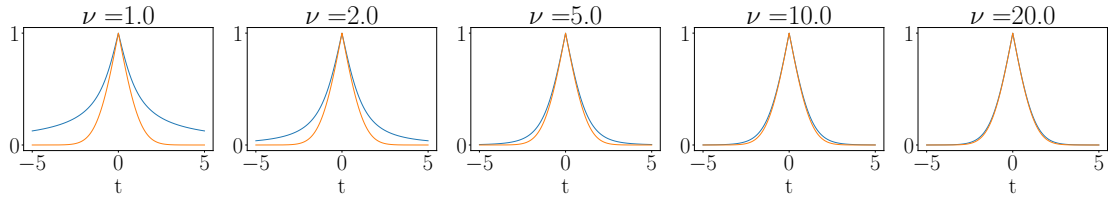


Figure 8.4: The true p-value p_ν for $\nu = 1, 2, 5, 10, 20$ (blue) and the approximation p_∞ based on the standard normal distribution (orange). As $\nu \to \infty$ the true p-value approaches p_∞.

8.3.3 Gaussian Approximation

For increasing degrees of freedom ν the t-distribution converges to the Gaussian

$$f_\infty(t) := \lim_{\nu \to \infty} f_\nu(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}}.$$

Figure 8.3 illustrates that the difference between f_ν and f_∞ vanishes quickly, even for "smaller" values of ν (e.g. 20).

As the evaluation complexity of p_ν increases with ν, we replace f_ν with f_∞ once ν is above a certain threshold and instead compute

$$p_\infty(t) := 2 \cdot \int_\infty^{-|t|} f_\infty(u)\, \mathrm{d}u = 1 - \operatorname{erf}\left(\frac{t}{\sqrt{2}}\right). \tag{8.3.5}$$

Figure 8.4 shows that the difference between p_∞ and p_ν also vanishes quickly, motivating this replacement. Since there exists no analytical solution for the above integral we use an approximation by Abramowitz and Stegun [144] with an error $\leq 2.5 \cdot 10^{-5}$:

$$\operatorname{erf}(t) \approx 1 - (0.3480242x - 0.0958798x^2 + 0.7478556x^3)e^{-t^2}\ ,\ x := \frac{1}{1 + 0.47047t}. \tag{8.3.6}$$

We did not detect any improvement when using one of their more complex approximations of erf. In our experiments (section 8.5.7) approximating p_ν with p_∞ whenever $\nu \geq 20$ has no noticeable impact on the test results.

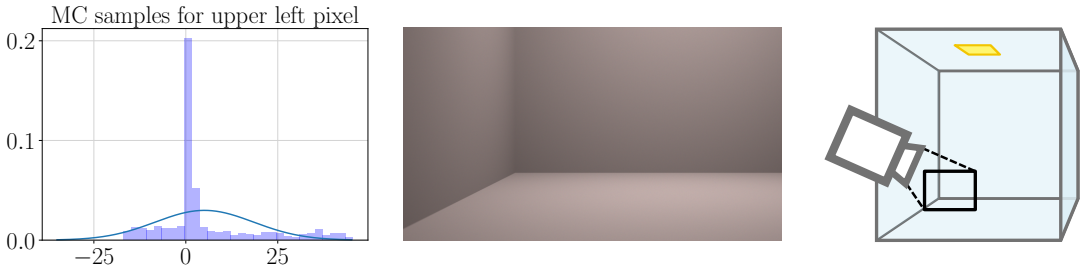


Figure 8.5: **Left:** Histogram of individual path traced Monte Carlo samples for the red channel of the upper left pixel in the corner scene (**center**: render, **right**: scene setup), and a fitted normal distribution (blue line).

8.4 Application to Monte Carlo Rendering

In this section we discuss how Welch's t-test can be applied to test a new renderer for unintended bias by comparing it to an unbiased reference implementation. We consider Monte Carlo renderers (section 4.5) that solve the *path integral*

$$I = \int_{\mathcal{P}} f(\mathbf{X}) \, \mathrm{d}\mathbf{X}$$

for each pixel numerically by sampling paths $\mathbf{X}_i$ from some probability distribution p over path space $\mathcal{P}$ and averaging their contributions $f(\mathbf{X}_i)$ to the pixel I in the estimator

$$F_N = \frac{1}{N} \sum_N \frac{f(\mathbf{X}_i)}{p(\mathbf{X}_i)} \approx I.$$

The expected value $\mathbb{E}[F_N]$ of the estimator (equation 4.3.2) is the mean μ of the underlying distribution of $\frac{f(\mathbf{X})}{p(\mathbf{X})}$. So given two implementations of unbiased estimators F_N^1, F_N^2 we will use Welch's t-test to test whether the underlying means of the implementations are equal. This would imply that their expected values are equal, which in turn means that they are both unbiased (assuming one of the implementations is a reference implementation known to be unbiased).

8.4.1 Generating Normal Distributed Samples

In order to conduct Welch's t-test we require two sets of normal distributed samples. However, the underlying distribution of a Monte Carlo estimator for light transport (i.e. the distribution of $\frac{f}{p}$) is typically not a normal distribution, as figure 8.5 illustrates.

The *central limit theorem* states that the normalized sum of an increasing number of samples drawn from some distribution tends to be normal distributed. We employ this theorem by adding up multiple samples generated by the Monte Carlo estimator, and using this sum as one sample for Welch's t-test.

For distinction, we use the term *Monte Carlo (MC) sample* to refer to one sample generated by the renderer for one pixel. For example, considering a path tracer with next event estimation (NEE), by "one MC sample" we mean the sum of MIS-weighted

contributions of one camera path and the NEE connections started from its inner vertices. In order to create one *Welch sample* X_i that can be used as input to Welch's t-test, we sum up a certain number of MC samples. If that number is large enough, the sum is roughly normal distributed.

More specifically, we conduct Welch's t-test separately for disjoint tiles of 32×32 pixels. One Welch sample is formed by summing up one MC sample from each pixel within that tile. This corresponds to stratified sampling of the underlying distribution of that pixel tile (by stratifying MC samples over pixels), and summing them up. For most test scenes used in our evaluation, summing up 1024 MC samples to get one Welch sample does indeed result in a (near) normal distribution of Welch samples within each pixel tile, as we evaluate empirically in section 8.5.1. Since we always sum up the same number of samples (1024) to get one Welch sample, we can skip the normalization (dividing the MC sample sum by 1024). If the image resolution is not a multiple of 32×32 we discard incomplete tiles. If a light tracer is used one needs to explicitly count the number of samples splatted into each tile per iteration and perform the normalization accordingly.

The central limit theorem does not offer any guarantees about a finite sum of samples approaching a normal distribution. So for each scene and algorithm, before applying Welch's t-test, one should evaluate the number of MC samples needed to (roughly) achieve a normal distribution of Welch samples. In section 8.5.1 we present one example where this is not the case for one MC sample per pixel in a 32×32 tile.

8.4.2 Sample Preparation during Rendering

As the renderer continues generating more samples per pixel, we can prepare sample statistics during rendering to use for Welch's t-test, as outlined by algorithm 5. For computing the t-statistic (equation 8.3.1) and the appropriate degrees of freedom ν (equation 8.3.3) we need the sample mean and sample variance of the Welch samples. Therefore the renderer needs to be able to create individual Welch samples by summing up one MC sample from each pixel within a pixel tile, in order to compute the sum of squares of Welch samples $\sum_N X_i^2$. We require two additional frame buffers with $1/32^2$ the resolution of the actual image: One for storing the sum of Welch samples for each pixel tile, and one for storing the sum of squared Welch samples. We compute and store the sum of Welch samples and their squares for each color channel separately.

Algorithm 5: Sample preparation during rendering

```
while Render Iteration do
    for each pixel block b do
        w = 0
        for each pixel p do
            x = render one sample for p
            w += x
        b.sum += w
        b.sumSquares += w*w
        N += 1 // count iterations
```

8.4.3 Considerations for Halton Points

In our implementation when using random numbers the renderer iterates over all pixels, places one random sample in each, and repeats. This means that for an image with N pixels, after placing N samples each pixel received exactly one sample, and therefore each pixel tile received 32^2 samples. We also support rendering with Halton points: In this case, pixel samples are not placed one for each pixel, but instead where the first two dimensions of the Halton points fall in the image plane: $(\Phi_2(i), \Phi_3(i))$, i.e. the radical inverse Φ with basis 2 and 3. We know that every 2×3 block will have received 6 samples after drawing N samples, however some pixels will have received more than one while others received none. This also means that in one iteration a 32×32 pixel tile may not always receive 32^2 samples.

The central limit theorem works on the normalized sum of samples, which means we would have to normalize the sum by the actual sample count, which is cumbersome. To arrive at a defined sample count per pixel, we make use of the elementary interval property of Halton points, which guarantees that after placing $3 \cdot N$ samples, each 2×2 block received 12 samples[1], and therefore each tile received $3 \cdot 32^2$ samples. We can then simply sum up those samples and skip normalizing. This issue can also be avoided by stamping repeated replications of the sample points for each tile.

Using quasi-Monte Carlo point sets introduces correlation which may yield inconclusive results when comparing two renders. To avoid this, we only compare renders with Halton points to ones with independent random points. Section 8.5.8 analyzes this.

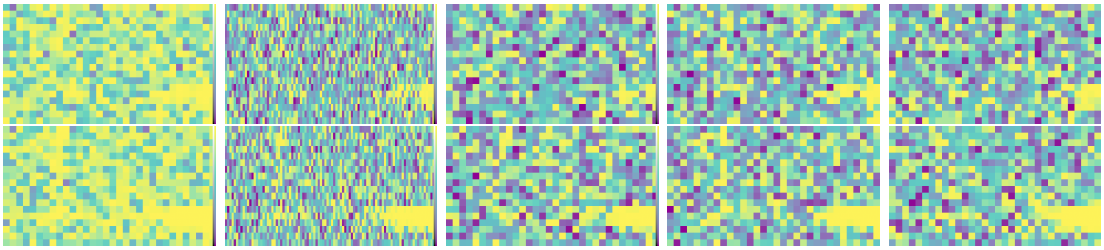
8.4.4 Visualization

We propose two visualizations for interpreting the results. In both cases we visualize the p-value, since it is bounded in $[0, 1]$, as opposed to the t-statistic which can take on arbitrary values.

p-value color map. As we conduct Welch's t-test separately for many image regions (i.e. square tiles of pixels) we can visualize the p-value obtained for each region in a color map. This scheme allows us to localize problematic regions in the image that have especially high bias. While this could also be achieved using difference images, Welch's t-test is both statistically meaningful and more expressive at low sample counts, as shown in section 8.5.3. Since we get three p-values for each image region (one per color channel) we try visualizing them combined and separately in figure 8.6. A color mapping scheme maps large p-values to violet and small ones to yellow, as is indicated to the right of each image, and enlarged in figure 8.7.

Visualizing only the smallest p-value per square pixel tile (figure 8.6 a), while skewing the image toward colors representing small values, seems easier to parse visually thanks to the square shape of individual p-value color tiles, and combines all values in a single image. Visualizing all three p-values requires us to use oblong blocks to squeeze three p-values into one square pixel tile (figure 8.6 b), or to output three separate images (figure 8.6 c-e). The former results in a cluttered image while the latter requires us to look at three separate images to evaluate the results.

[1]This only holds for image resolutions $2^j \times 3^k$, otherwise the pixel grid and the elementary intervals do not align. In our experiments this still seemed to be a sufficiently good approximation.

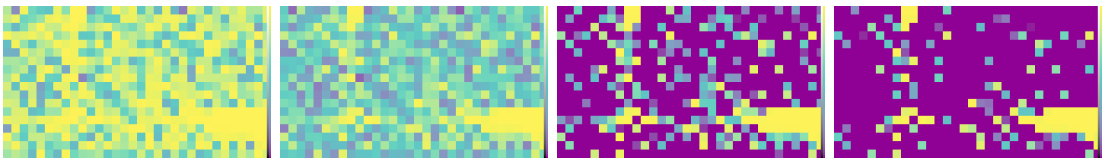


(a) smallest (b) all values (c) red channel (d) green channel (e) blue channel

Figure 8.6: Color map for Welch's t-test applied to 32×32 pixel tiles of an unbiased PT render and a biased BDPT (as in section 8.5.3) render of the corner scene (figure 8.10 left) with 10 spp (**top**) and 60 spp (**bottom**), visualizing the smallest (a), all three (b) and individual p-values (c,d,e) per color channel.



Figure 8.7: Color scale for $p = 0$ (left) to $p = 1$ (right).



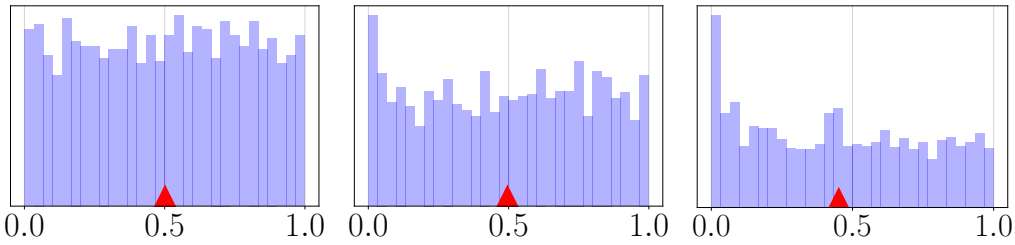
(a) original (b) square root (c) multiplied by 10 (d) multiplied by 20

Figure 8.8: Different p-value mappings (**left**: Same as figure 8.6).

One advantage of the last visualization is that it reveals whether clusters of small p-values just happened at random, or are symptomatic of some intrinsic problem, i.e. present in all three color channels. In general, we find the visualization of the smallest p-value easiest to use. In the example in figure 8.6 it best reveals the bias along the left corners of the scene.

Figure 8.8 visualizes the square root or a multiple of the p-value, in order to focus a higher color resolution on small p-values corresponding to potentially biased regions.

Histogram of p-values. Whenever the null hypothesis (the two compared renderers being unbiased) is true, the p-value, which is a random variable itself, is uniformly distributed (see section 8.3.2). Since we conduct only one test per color channel and image region, each corresponding to their own distribution, we cannot directly check whether the p-value obtained for one image region is uniformly distributed. However, we can at least check whether the p-values taken from different regions are, based on a histogram of all p-values, as well as their mean. A uniform histogram is no proof that all individual p-values stem from uniform distributions. However, a non-uniform histogram is a strong indicator that the individual p-values cannot all be uniformly distributed and thus the tested renderer is likely biased. Three examples are shown in figure 8.9.



(a) unbiased PT (b) biased BDPT, 10 spp (c) biased BDPT, 60 spp

Figure 8.9: Histogram of p-values for an unbiased render (a) and two biased renders (biased BDPT as in section 8.5.3 with 10 (b) and 60 (c) samples per pixel (spp), same as in figure 8.6) tested against an unbiased reference implementation. At 10 spp the p-value mean (red triangle) is indistinguishable from 0.5, but the $[0, 1/30]$-bin contains way more p-values than would be expected if all p-values were uniformly distributed, which indicates bias.



Figure 8.10: **Left**: Corner scene. **Right**: Dining scene.

8.5 Results

We evaluate Welch's test for two scenes, a simple scene showing the lower left corner of an empty Cornell box with a constant square light source (figure 8.10 left), and the dining scene (figure 8.10 right). The first scene is helpful in detecting basic errors in the transport algorithm itself, e.g. problems with the camera or missing cosines. The second scene illustrates the test's behavior under more complex conditions. We use a path tracer with next event estimation (PT) and a bidirectional path tracer (BDPT) in our custom spectral renderer.

Unless stated otherwise, the renderers draw their random numbers from a pseudo-random number generator, and Welch samples are constructed within 32×32 pixel tiles by summing up one sample from each pixel within the tile.

8.5.1 Welch Sample Generation

Before applying Welch's t-test, we need to make sure that the number of MC samples contributing to one Welch sample is sufficient for (approximating) a normal distribution of Welch samples, as described in section 8.4.1. We therefore generate histograms of the Welch samples created for individual tiles within the two images in figure 8.11.

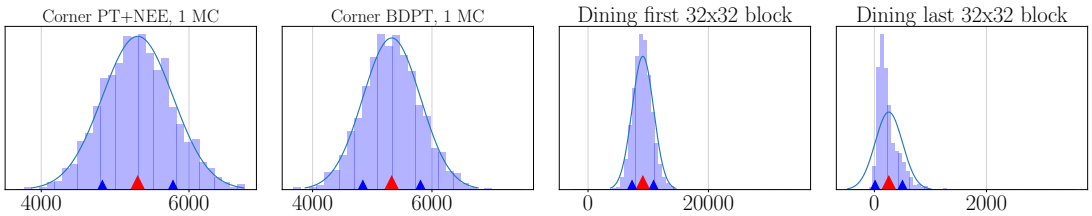
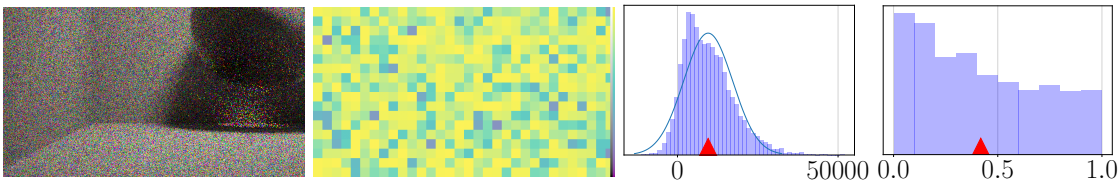


Figure 8.11: Histogram of Welch samples generated by summing up one MC sample per pixel for the red channel of a 32×32 pixel tile. **Left, center left**: Upper left pixel tile in figure 8.10. **Center right**: Upper left pixel tile in the dining scene rendered with PT. **Right**: Lower right pixel tile in the dining scene (PT). MC samples are based on Mersenne random numbers. The red and blue markers represent the sample mean and variance, the blue line is a fitted normal distribution using those as mean and variance.

For simple setups, such as the corner scene or the directly lit upper wall in the dining scene, summing up 1024 MC samples per pixel tile creates normal distributed samples for both the path tracer and the bidirectional path tracer. However, in more difficult setups, such as the caustic on the table or the shadowed lower right part of the wall, Welch samples created that way are not as close to a normal distribution. It turns out that Welch's t-test still produces reasonable results for the dining scene with Welch samples created that way, but in general one should be cautious about the number of MC samples contributing to one Welch sample.

Failure case. Figure 8.12 shows the corner scene with a glass sphere added in the center. The resulting caustic noise causes the Welch samples in the respective image regions to not be normal distributed anymore, even for larger tiles such as 64×64 pixels as shown in figure 8.12c. Therefore Welch's t-test should not be used with this scene, as it produces a far from uniform histogram of p-values even when testing the reference implementation against itself with different seeds for the random number generator. If a renderer produces a Welch sample distribution such as this one, one might reconsider using that algorithm for the given scene, since its convergence will be rather slow anyway.



(a) scene (b) Color map (c) Welch samples (d) p-values

Figure 8.12: The corner scene with a glass sphere no longer has normal distributed Welch samples when rendered by a path tracer, not even when summing up 64×64 pixel samples as in this example (c). Thus, when testing the unbiased reference against itself (shown here for 10 spp) Welch's t-test does not produce a uniform histogram of p-values (d) anymore.

8.5.2 Unbiased Reference Test

We start with the test results for a correct path tracer with next event estimation tested against itself in the corner scene (figure 8.10 left) with different random seeds. This serves as a reference for what the results should look like for unbiased renders. Figure 8.13 shows the color map and p-value histogram for testing 10 and 1000 spp against 10 and 1000 spp renders.

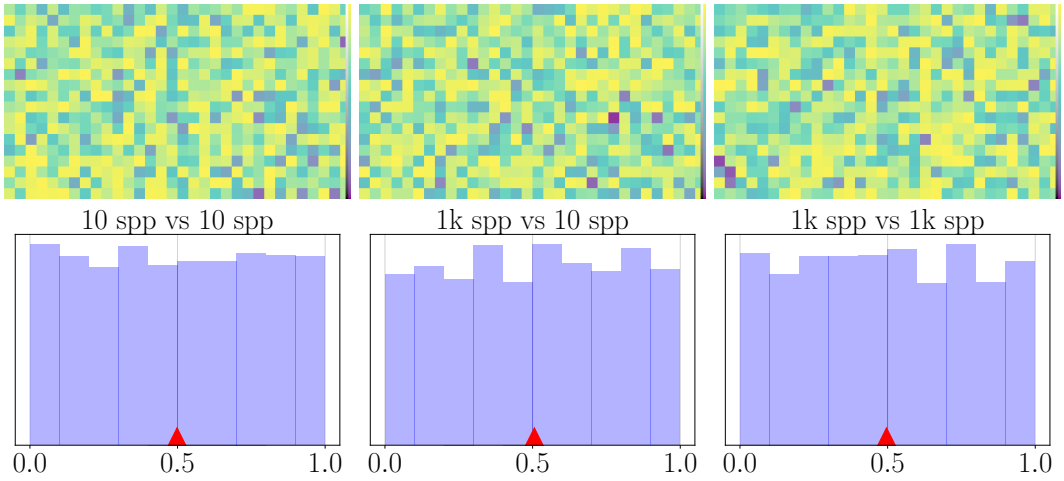


Figure 8.13: Test results of an unbiased path tracer against itself for the corner scene (figure 8.10 left). **Left to right**: 10 spp vs 10 spp, 10 spp vs 1000 spp, 1000 spp vs 1000 spp. **Top**: Color map for smallest p-value. **Bottom**: Histogram (blue) and mean (red) of p-values from all color channels and pixel tiles.

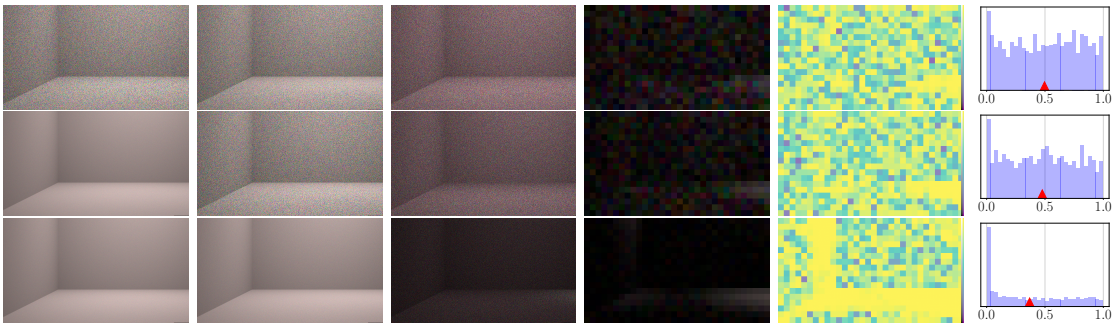
8.5.3 Finding Bias in a Bidirectional Path Tracer

We used Welch's t-test to trace down a bug in our bidirectional path tracer (BDPT). In this version, the light tracer occasionally replaced the camera normal with the connecting segment's direction, leading to an incorrect cosine evaluation between the segment's direction and camera normal. This affected the light tracer's contribution as well as the MIS-weight computation. The resulting error was stronger toward the borders of the image, where the correct cosine is much smaller than the cosine that was computed (which was always one).

Figure 8.14 shows the PT reference and the biased BDPT renders, both with 3-vertex paths only, as well as the results of Welch's t-test and difference images. The clumped yellow regions in the color maps are strong evidence that the two renderers do not converge to the same mean, i.e. pixel color, at least not in those image regions. Similarly, the non-uniform histogram indicates that not all, maybe even none, of the p-values stem from a uniform distribution, and thus one of the renderers is biased.

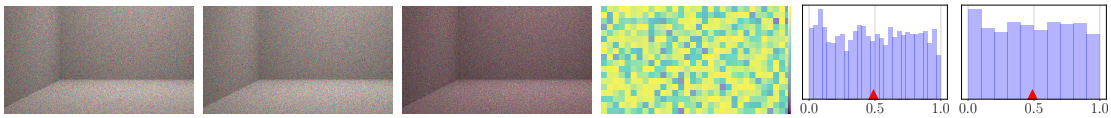
In this case, using more samples in the biased renderer results in even more significant evidence (more low p-values, manifesting as clumped yellow regions and a non-uniform histogram) against the hypothesis that the renderer is unbiased.

For a fair comparison to what insight can be gained from difference images, we also include a tile-wise difference image, such that one difference value can benefit from the same number of MC samples as one t-test.



(a) PT (b) BDPT (c) Difference (d) Block diff (e) Color map (f) p-values

Figure 8.14: **Top**: PT (10 spp) (a) vs biased BDPT (10 spp) (b). Differences (c, d) are absolute.
Center: PT (1k spp) vs biased BDPT (10 spp) shows that a more converged reference produces a visually similar color map, but a more conclusive p-value histogram.
Bottom: PT (1k spp) vs biased BDPT (500 spp). Only now do we start to get a meaningful difference image (c, d), revealing a subtle difference in brightness at the right part of the floor.



(a) Gray=0.2 (b) Gray=0.21 (c) Difference (d) Color map (e) p-values (f) p-values

Figure 8.15: Corner scene with gray = 0.2 (a) and gray = 0.21 (b), PT, 10 spp. The absolute difference (c) is dominated by noise and at this sample count is indistinguishable from a difference image of two correct renders with different seeds. In this case the histogram of p-values with fewer bins (f) is most helpful at detecting bias.

8.5.4 Detecting a Biased Scene

In figure 8.15 we show two renders of the box scene created by the unbiased PT with 10 spp. On the left, the walls and floor have the diffuse color RGB = (0.2, 0.2, 0.2) while on the right they have color RGB = (0.21, 0.21, 0.21). In this case the color map and difference image are useless, the latter being dominated by residual noise. Yet, the p-value histogram reveals a tendency toward small values, indicating that the two renders will not converge to the same result. With fewer bins (f) this is even more noticeable.

8.5.5 Biased MIS Weights in a Path Tracer

Figure 8.16 shows Welch's t-test for a path tracer with incorrect multiple importance sampling (*MIS*) weights, where for paths finding the light source by accident the probability density function of the last scattering event is ignored in the MIS weight computation. This results in a slightly brighter image, which is hardly noticeable by eye (see figure 8.17 (a)). In this case, Welch's t-test does not pick up on the error at 10 spp, but is able to detect it at much higher sample counts. As the difference image and color map look similarly unhelpful as in figure 8.15 we only show the p-value histograms at different sample counts in figure 8.16.

Since in this case Welch's t-test requires high sample counts, we also compare to the RMSE plot in figure 8.17. We only plot the RMSE up to 8k samples, since our reference has 131k samples and plotting for larger sample counts would result in a curve caused by the unconverged reference. For instance, when using a 65k reference instead, this effect is already noticeable at a few thousand samples per pixel (figure 8.17 (c)). With the 131k spp reference, the RMSE plot is a straight line and does not reveal any error. Computing a reference with even more samples to eventually detect a curving RMSE line would be possible but impractical.

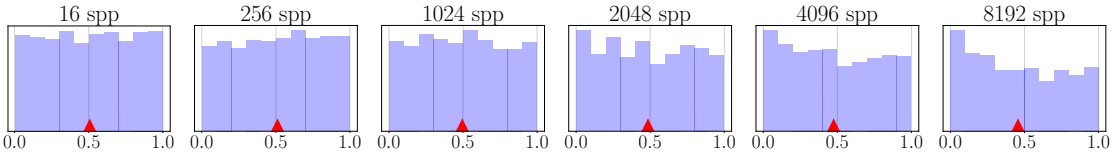
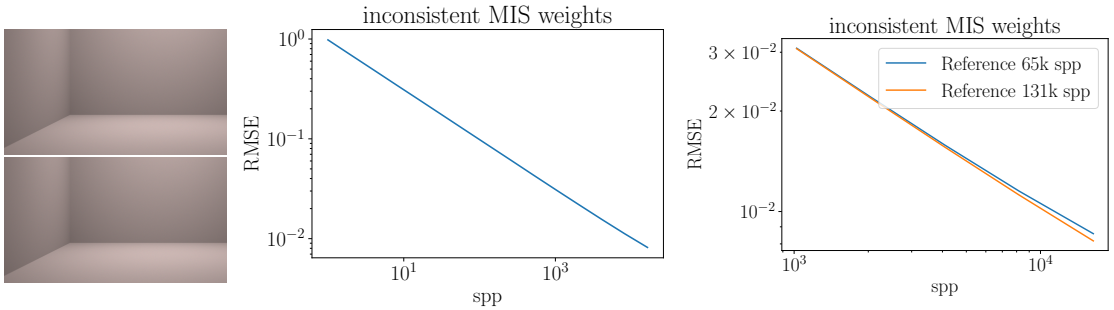


Figure 8.16: p-values for the path tracer with biased MIS weights. Welch's t-test only detects the bias reliably at several thousand samples per pixel, where it produces the non-uniform histograms.



(a) PT (b) RMSE (c) RMSE (zoomed)

Figure 8.17: (a) The corner scene rendered with a path tracer at 131072 (2^{17}) spp, with reference PT (**top**) and biased PT due to inconsistent MIS weights (**bottom**). The RMSE plot (b) looks like a straight line and thus does not reveal any bias. With this reference, any curve occurring at higher sample counts might as well be caused by the unconverged reference, as is already the case when plotting the RMSE with a 65k spp reference ((c), zoomed in).

8.5.6 Influence of the Pixel Tile Size

Figures 8.18 and 8.19 show the test's result if we use individual MC samples as Welch samples and conduct Welch's t-test per pixel instead of per pixel tile. Figure 8.18 tests the unbiased PT reference against itself, where the test produces a false reject (a non-uniform histogram of p-values) in case of the dining scene. Figure 8.19 tests the unbiased PT against the biased BDPT described in section 8.5.3 for the corner scene. Here the test does not detect bias at 60 spp, and instead outputs a uniform color map and histogram of p-values, producing a false positive. We conclude that Welch's t-test is unreliable both in detecting bias and in not-rejecting unbiased renderers when using MC samples directly as Welch samples.

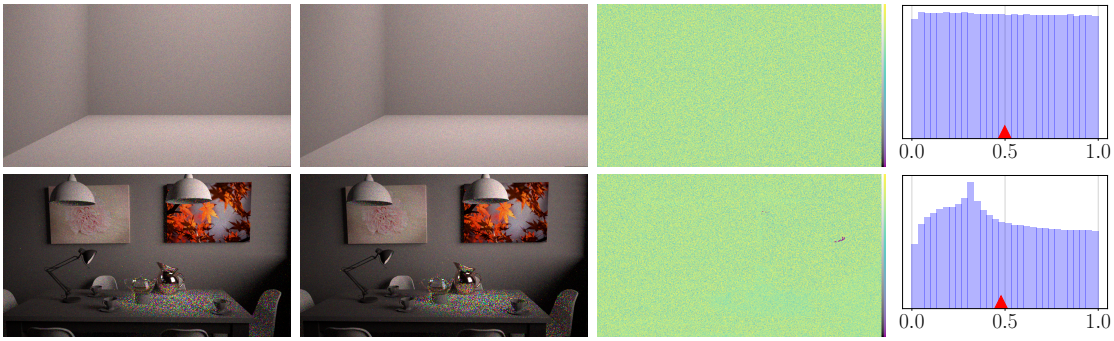


Figure 8.18: Testing the unbiased PT against itself with different random seeds and 100 spp each for two different scenes. Welch's t-test is conducted per pixel, and individual MC samples are used as input samples to the test.

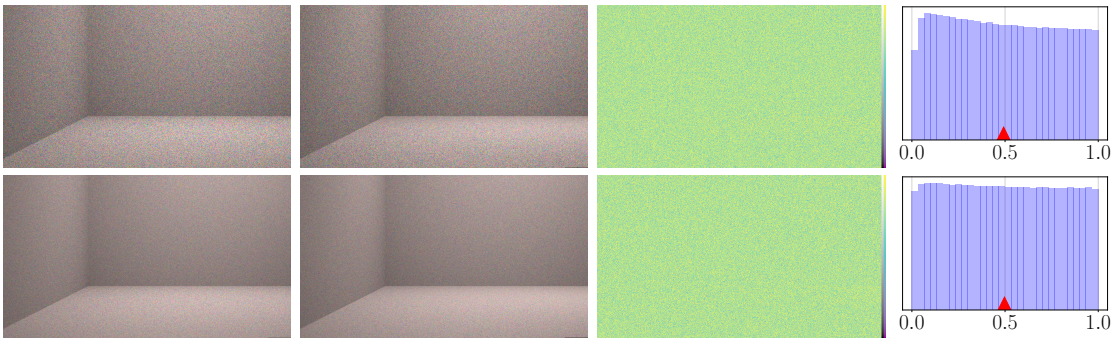


Figure 8.19: Testing the unbiased PT (**left**) against a biased (section 8.5.3) BDPT (**center left**), resulting color map for smallest p-value (**center right**) and respective p-value distributions (**right**). Welch's t-test is conducted per pixel, and individual MC samples are used as input samples to the test. **Top**: 20 spp. **Bottom**: 60 spp.

8.5.7 Gaussian Approximation

Figure 8.20 shows the range of appropriate degrees of freedom ν (equation 8.3.3) over all image regions when comparing different sample counts for the corner and dining scene. Starting from a few dozen Welch samples we consistently get ν values such that the normal distribution approximates the actual t-distribution quite well over the entire image. With fewer Welch samples, this is only the case in some regions, while other regions have low ν-values, as each region corresponds to its own distribution.

Figure 8.21 shows p-values p_ν computed from f_ν with the correct degrees of freedom ν, as well as p_∞ based on the approximated (see equations 8.3.5 and 8.3.6) normal distribution f_∞. If the number of samples is not sufficient (e.g. $N_1 = 10$, second column) and the appropriate ν is too small, the approximated p-values are not uniformly distributed, leading to false negatives (Welch's t-test "finding" bias when comparing two unbiased renderers). This is to be expected, as the true p-value is larger than the Gaussian approximated p-value (see figure 8.4 left), and thus the Gaussian approximation results in more p-values close to zero than a uniform distribution should have. This also explains why the Gaussian approximation only creates more false negatives, but no more false positives than p-values based on the true f_ν.

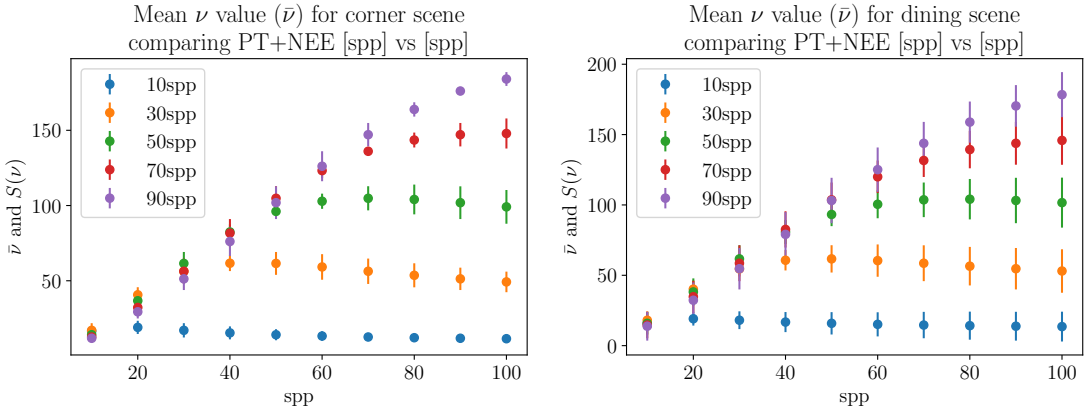


Figure 8.20: Mean ($\bar{\nu}$) and standard deviation ($S(\nu)$) of ν-values over all pixel tiles when comparing our PT against itself for different sample counts for the corner (**left**) and dining scene (**right**), with 1 sample per pixel per 32×32 tile per Welch sample. The ν-value seems to depend mostly on the lower sample count and less on the scene.

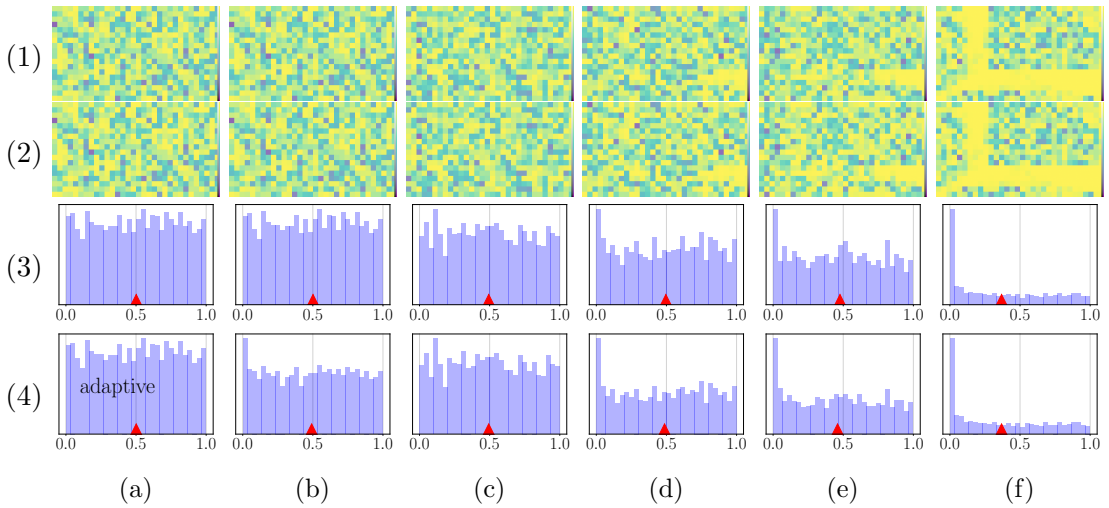


Figure 8.21: Welch's t-test for the corner scene with the correct t-distribution ((1),(3)), compared to the Gaussian approximation as in equation 8.3.5 ((2),(4)).
In (a) the approximation is applied wherever $\nu \geq 20$, in (b)-(f) it is applied to all image regions and color channels.
(a), (b) PT (10 spp) vs PT (10 spp) ($\bar{\nu} \approx 20$)
(c) PT (10 spp) vs PT (1k spp) ($\bar{\nu} \approx 2000$)
(d) PT (10 spp) vs biased BDPT (10 spp) ($\bar{\nu} \approx 20$)
(e) PT (1k spp) vs biased BDPT (10 spp) ($\bar{\nu} \approx 10$)
(f) PT (1k spp) vs biased BDPT (500 spp) ($\bar{\nu} \approx 1000$)
While the color maps seem indistinguishable, the histogram of approximated p-values for (b) ($\bar{\nu} \approx 20$) falsely rejects the hypothesis of the renderers being unbiased. Selectively applying the approximation in tiles and color channels with $\nu \geq 20$ fixes this (a).

Selectively applying the Gaussian approximation only in image regions and color channels where $\nu \geq 20$ fixes this, as shown in the left column in figure 8.21. Here ($N_1 = 10$) the value of ν still varies over image regions and channels. For many the approximation is valid, but the few where it is not, due to low ν-values, it results a non-uniform histogram when using the approximation.

8.5.8 Halton Points

If we use Halton points to place MC samples in the pixel plane, we need to sum up a multiple of 3 samples per pixel to assure an equal number of MC samples per Welch sample (section 8.4.3). Figure 8.22 (below) shows a histogram of Welch samples constructed this way on the right, and a histogram of Welch samples created from *on average* 32^2 MC samples per tile on the left. At least in this example, both approaches yield similarly distributed Welch samples.

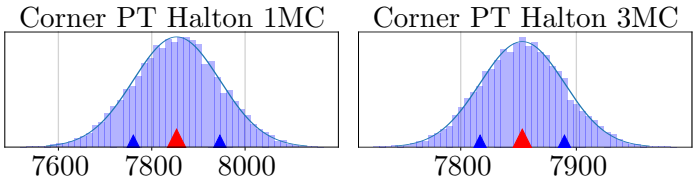


Figure 8.22: 10k Welch samples for the lower right 32×32 pixel tile in the corner scene for a path tracer based on Halton points. **Left**: On average 1 MC sample per pixel summed up for 1 Welch sample. **Right**: 3 MC samples per pixel per Welch sample.

We show the test's results based on such samples, compared against a Mersenne twister based reference, of the unbiased PT in figure 8.23 and of the biased BDPT (section 8.5.3) in figure 8.24. We did not properly normalize the Welch samples according to the number of MC samples that were actually used to create them, and assume a constant normalization instead. Yet, Welch's t-test seems to work regardless of the Welch sample distribution less resembling a normal distribution than when using 3 MC sample per pixel per Welch sample, and the tests yield qualitatively similar results.

Correlation. Figure 8.25 reveals another issue: Due to the correlations in the Halton dimensions, i.e. those used to place MC samples in the pixel plane, testing two renders both of which are based on Halton points yields useless results. When testing a Halton based unbiased BDPT against either unbiased Halton based PT or BDPT, the corner regions look at least a bit as expected, likely due to the larger influence of the light tracer which does not depend on an explicit sample in the pixel plane.

8.5.9 Run Time and Memory Consumption

It took roughly 3 seconds on an Intel i7-6700 to run 110 tests with $N_1, N_2 \in [10, 100]$, half of which used the Gaussian approximation, with image resolution 1024×576 and tile size 32×32. Therefore we did not consider further optimizations or faster approximations, especially since in a typical use case one would conduct only one test at a time.

As for memory consumption, we store two additional frame buffers during rendering with $1/32^2$ the resolution of the original frame buffer. With sample set sizes $N_{1,2}$ up to the order of 10^3 we found 32-bit floating point numbers to be sufficiently precise.

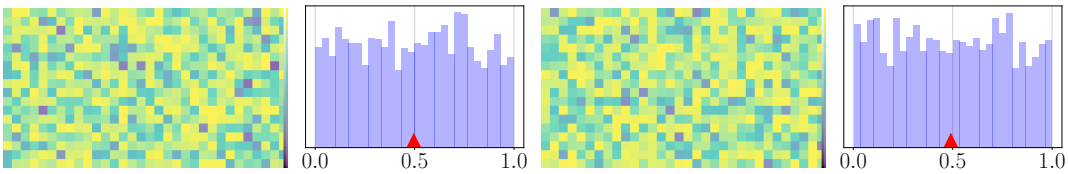


Figure 8.23: Testing the unbiased PT based on Halton points in the corner scene against a PT based on Mersenne random numbers. **Left**: On average 1 MC sample per pixel summed up for 1 Welch sample, 10 spp. **Right**: 3 MC samples per pixel per Welch sample, 30 spp. Both yield fairly uniform histograms.

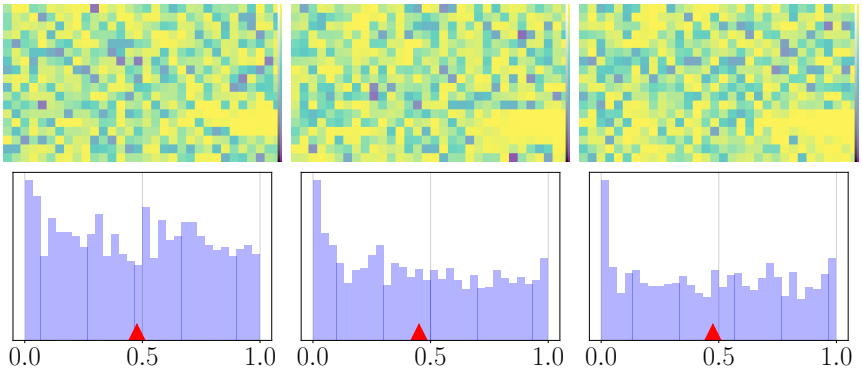


Figure 8.24: Testing the biased BDPT based on Halton points in the corner scene. **Left**: on average 1 MC sample per pixel summed up for 1 Welch sample, 10 spp. **Center**: 3 MC samples per pixel per Welch sample, 30 spp. Both cases have 10 Welch samples available per test; if we sum up on average 1 MC sample per pixel but use 30 spp (**right**), the test achieves similar, if not slightly better, quality.

8.6 Conclusion

We showed how to apply Welch's t-test to test supposedly unbiased renderers for bias compared to a reference implementation. Welch's t-test can help locate faulty image regions even at low sample counts, and detects bias much earlier than RMSE or difference image tests. Our color visualization proved useful at identifying particularly biased image regions, while at very low sample counts the p-value histogram revealed errors that were not yet obvious in the color map, a difference image or RMSE plot.

Our results suggest that Welch's t-test is more reliable if it is applied to (approximately) normal distributed samples. Therefore, before using Welch's t-test, depending on the scene, one should make sure to use an appropriate pixel tile size and sample-per-pixel count within tiles to ensure at least roughly normal distributed Welch samples. When in doubt, it may help to first test the reference against itself, and if the test fails to produce a uniform histogram, fall back on some other method.

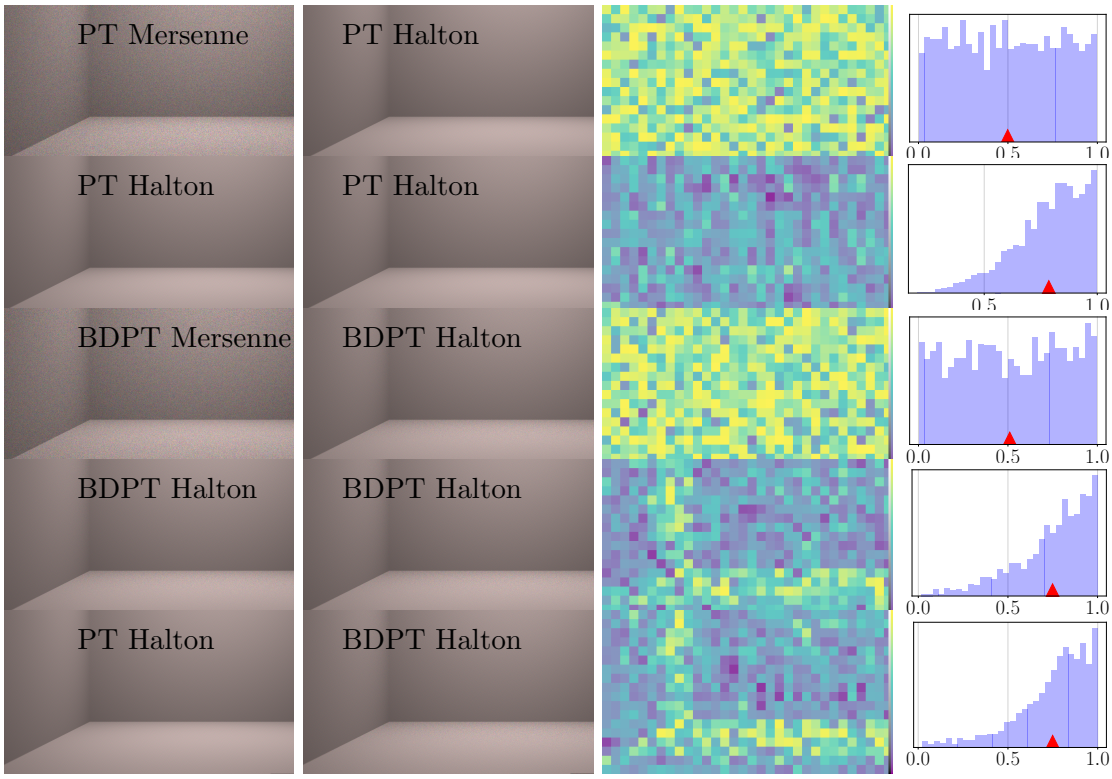


Figure 8.25: Unbiased PT and unbiased BDPT in the corner scene. Each image has 60 MC samples per pixel and different random seeds. The test uses 20 Welch samples with 3 MC samples per pixel per 32×32 tile. Whenever both renders are based on Halton points, Welch's t-test produces misleading results.

Limitations

Welch's t-test only tells us whether two distributions might have the same mean. Therefore we can only use it to detect incorrect rendering algorithms when compared to a reference implementation that is known to work. We cannot use it to assess asymptotic error behavior, i.e. we cannot use it to tell which of two algorithms converges faster.

Large p-values are no proof that two renderers converge to the same image, neither is a uniform histogram of p-values; we can only use a big number of pixel tiles with small p-values (e.g. < 0.01) as strong evidence that they do not. Since we apply Welch's t-test to many image tiles at once, we can use all the individual results at once and thus at least decrease the chance of Welch's t-test not finding existing bias.

Although it did work in most of our test cases, due to the numerical nature of the problem, there is no guarantee that Welch's t-test always detects existing bias at a low sample count. In one example of particularly small bias, the test only detected bias at a few thousand samples per pixel, which however was still less than what was required to detect the bias using the RMSE plot.

9 Conclusion

In the context of time constrained rendering we considered regularization for faster initial convergence to acceptable image quality, automated conversion of RGB input data for faster scene design, and Welch's t-test for quick validation iterations.

Our approach to upsample RGB input data to a combination of non-fluorescent and fluorescent spectra is based on a precomputed 3D lookup table. This enables a fast conversion of existing RGB textures to our coefficient space. Remaining issues with interpolation across the boundary between RGB values that do and do not require fluorescence need to be addressed for better robustness in practice.

In scenes with fluorescent input data we showed that conventional non-fluorescent surfaces behave as spectral singularities, just like surfaces such as mirrors or glass present geometrical singularities in classical rendering. This insight allowed us to apply existing mollification techniques to these spectral singularities. We can control the regularization intensity to choose between fast initial convergence at the cost of higher bias, particularly under tight time constraints, and lower long-term bias in exchange for higher initial variance.

Scenes with (near) specular surfaces and (near) singular light sources pose problems to unregularized path tracing-based approaches. Techniques that blindly apply regularization to all scene features such as photon mapping may resolve such configurations, but also apply bias to other features where it is not needed. More elaborate approaches such as vertex connection and merging introduce unnecessary complexity when applied to otherwise less demanding scenes. Our selective regularization for connecting (near) specular chains to small area lights avoids unnecessary bias in unproblematic image features. We balance regularized with unregularized contributions by considering the relative size of the light sources and the regularization kernel.

In summary, the approaches presented in this thesis allow for a more robust and efficient light transport simulation in scenes with difficult integration problems and fluorescent materials, which are crucial for accurate predictive image synthesis. The desire for increasingly high realism means we need to rethink traditional material models, to include not only fluorescence but also other phenomena. Considering the increasing complexity of scene data in the industry, both data-driven approaches (such as [9], [10] and [11]) and regularization of particularly problematic features (chapters 6 and 7) promise higher robustness in the face of various input configurations, without requiring manual modifications or impractically high sample counts.

A Appendix: Background

A.1 Light Transport as a Measure

In this section we show how to express singular light transport components based on the Dirac measure (equation 4.1.1)

$$\delta_\omega(\mathcal{D}) = \begin{cases} 1 & \text{if } \omega \in \mathcal{D} \\ 0 & \text{else,} \end{cases}$$

where

$$\int_D \Phi(x)\,\mathrm{d}\delta_\omega(x) = \Phi(\omega).$$

We also construct measures corresponding to the BSDF or incident radiance function for non-singular components.

In particular, we will move the scattering behavior and incident light to a new measure $\bar{\mu}^{\mathbf{x},\omega_o}$ such that

$$L_o(\mathbf{x},\omega_o) = \int_{\mathbb{S}^2} \mathrm{d}\bar{\mu}^{\mathbf{x},\omega_o}(\omega_i).$$

The new measure $\bar{\mu}^{\mathbf{x},\omega_o}$ combines the solid angle measure σ and new measures $\bar{f}_r, \bar{L}_i$ for the surface scattering and incident light. To derive this new measure we first represent either the scattering behavior or the incident light with separate measures $\bar{f}_r$ and $\bar{L}_i$, before presenting the combined measure $\bar{\mu}$.

Scattering measure. The scattering measure of a surface is either based on the surface BSDF for non-singular surfaces, or on the Dirac measure[1] for specular surfaces with albedo $a(\omega_o)$:

$$\bar{f}_r^{\mathbf{x},\omega_o} = \begin{cases} \bar{f}_{BSDF}^{\mathbf{x},\omega_o} & \text{if } \mathbf{x} \text{ non-specular} \\ a(\omega_o)\delta_{\omega_{or}} & \text{else.} \end{cases}$$

We construct the scattering measure $\bar{f}_{BSDF}^{\mathbf{x},\omega_o}$ of a non-singular surface from the surface's BSDF as

$$\bar{f}_{BSDF}^{\mathbf{x},\omega_o} : \mathcal{B}(\mathbb{S}^2) \to \mathbb{R}^+,$$
$$\bar{f}_{BSDF}^{\mathbf{x},\omega_o}(\mathcal{D}) = \int_{\mathcal{D}} f_r(\omega_i,\mathbf{x},\omega_o)\cos\theta_i\,\mathrm{d}\sigma(\omega_i).$$

[1] Note that the $\delta_{\omega_{or}}$ in this expression is an actual measure, and not the symbolic shorthand notation $\delta^\sigma_{\omega_{or}}$ from the previous section.

Integrating an incident radiance function[2] over $\mathbb{S}^2$ with this measure results in the original rendering equation

$$\int_{\mathbb{S}^2} L_i(\mathbf{x}, \omega_i) \, \mathrm{d}\bar{f}_{BSDF}^{\mathbf{x},\omega_o}(\omega_i) = \int_{\mathbb{S}^2} L_i(\mathbf{x}, \omega_i) f_r(\omega_i, \mathbf{x}, \omega_o) \cos\theta_i \, \mathrm{d}\sigma(\omega_i),$$

and inserting the same radiance with the Dirac measure results in

$$\int_{\mathbb{S}^2} L_i(\mathbf{x}, \omega_i) \, \mathrm{d}[a(\omega_o)\delta_{\omega_{or}}](\omega_i) = L_i(\mathbf{x}, \omega_{or}) a(\omega_o).$$

Singular light measure. Similarly, the incident light measure is a placeholder for a Dirac measure for point lights, and a radiance-based measure for area lights:

$$\bar{L}_i(\mathcal{D}) := \begin{cases} \frac{I_p}{||\mathbf{p}-\mathbf{x}||^2}\delta_{\omega_p} & \text{for a point light at } \mathbf{p} \text{ in direction } \omega_p \\ \int_{\mathcal{D}} L_i(\mathbf{x}, \omega_i) \, \mathrm{d}\sigma(\omega_i) & \text{else.} \end{cases}$$

Integrating a BSDF and cosine with this measure results in

$$\int_{\mathbb{S}^2} f_r(\omega_i, \mathbf{x}, \omega_o) \cos\theta_i \, \mathrm{d}\bar{L}_i^{\mathbf{x},\omega_o}(\omega_i) = \begin{cases} \frac{I_p}{||\mathbf{p}-\mathbf{x}||^2} f_r(\omega_p, \mathbf{x}, \omega_o) \cos\theta_p & \text{point lights} \\ \int_{\mathbb{S}^2} L_i(\mathbf{x}, \omega_i) f_r(\omega_i, \mathbf{x}, \omega_o) \cos\theta_i \, \mathrm{d}\sigma(\omega_i) & \text{else.} \end{cases}$$

Combined measure. We can pull both the incident light and the scattering behavior in a combined measure $\bar{\mu}^{\mathbf{x},\omega_o}$, as long as at most one of them is singular:

$$\bar{\mu}^{\mathbf{x},\omega_o} = \begin{cases} \int_{\mathcal{D}} L_i(\mathbf{x}, \omega_i) f_r(\omega_i, \mathbf{x}, \omega_o) \cos\theta_i \, \mathrm{d}\sigma(\omega_i) & \text{if neither is singular,} \\ L_i(\mathbf{x}, \omega_{or}) a(\omega_o) \delta_{\omega_{or}}(\mathcal{D}) & \text{if the surface scattering is singular,} \\ \frac{I_p}{||\mathbf{p}-\mathbf{x}||^2} f_r(\omega_p, \mathbf{x}, \omega_o) \cos\theta_p \delta_{\omega_p}(\mathcal{D}) & \text{if the incident light is singular.} \end{cases}$$

However, if both the light source and surface scattering are singular (e.g. a point light seen through a mirror), this approach no longer works, at least not as long as we only consider the outgoing radiance at $\mathbf{x}$. This is because in such a scenario, there exists no outgoing radiance L_o at $\mathbf{x}$, similar to how there exists no outgoing radiance at a point light. We discuss this case in more detail in chapter 7.

[2] As discussed in the main thesis, if the incident radiance is singular (e.g. from (specularly reflected) point lights), there is no function that we can insert for L_i to get the desired behavior.

B Appendix: Spectral Mollification for Bidirectional Fluorescence

B.1 Energy Conservation

In this section we show that r_{mi}

$$r_{mi}(\lambda_i \to \lambda_o) = \frac{1}{2d} r(\lambda_o) \mathbb{1}_{|\lambda_i - \lambda_o| < d}$$

(mollification over incident wavelengths as done in equation 6.3.2) is not necessarily energy conserving when applied to the diagonal of a physically valid reradiation matrix, or to the full non-fluorescent term

$$\delta_{\lambda_o}(\lambda_i)(1 - ca(\lambda_i))r(\lambda_i) \tag{B.1.1}$$

of the BBRRDF by [8]. For example, consider

$$c = Q = 1, \lambda_i = 600, d = 50$$

and the extreme case of a box function for an absorption spectrum

$$a(\lambda_i) = \begin{cases} 1 & \text{if } \lambda_i \in [500, 600] \\ 0 & \text{else} \end{cases}$$

and a reflectance spectrum

$$r(\lambda) = 1 - a(\lambda).$$

In this case r_{mi} is not energy conserving:

$$\begin{aligned}
\int_\Lambda r_{mi}(\omega_i, \lambda_i, \omega_o, \lambda_o) \, \mathrm{d}\lambda_o &= \int_\Lambda \mathbb{1}_{|\lambda_i - \lambda_o| < d} \cdot \frac{(1 - ca(\lambda_o))r(\lambda_o)}{2d} + ca(\lambda_i)Qe(\lambda_o) \, \mathrm{d}\lambda_o \\
&= \frac{1}{2d} \int_{\lambda_i - d}^{\lambda_i + d} (1 - ca(\lambda_o))r(\lambda_o) \, \mathrm{d}\lambda_o + 1 \\
&= \frac{1}{2d} \int_{\lambda_i - d}^{\lambda_i + d} r(\lambda_o) \, \mathrm{d}\lambda_o - \frac{c}{2d} \int_{\lambda_i - d}^{\lambda_i + d} a(\lambda_o)r(\lambda_o) \, \mathrm{d}\lambda_o + 1 \\
&= \frac{1}{2d} \int_{\lambda_i - d}^{\lambda_i + d} r(\lambda_o) \, \mathrm{d}\lambda_o + 1 \\
&= \frac{1}{2d} \cdot d + 1 > 1.
\end{aligned}$$

Even though truly box-like spectra do not appear in the real world, this serves to show that in general, whenever the absorption and emission spectrum peak in adjacent areas, energy conservation might be an issue when using f_{mi}. The effect is visualized in figure B.2 (left) with a reradiation matrix constructed from measured reflectance and fluorescence spectra (figure B.1). Figure B.2 (right) shows what happens for constant mollification with f_{mo} instead, which does not break energy conservation.

Another way to think about mollification in the context of reradiation matrices is to pick some λ_i and view f_{mi} as an additional fluorescent component with $a(\lambda) = \mathbb{1}_{\lambda=\lambda_i}$ and $e(\lambda) = \frac{\mathbb{1}_{|\lambda-\lambda_i|<d}}{2d}$. Mollifying with f_{mi} would then correspond to adding the reradiation matrix of the original fluorescent component to the reradiation matrix of the mollified component. Depending on how extreme the original matrix was, this can now result in columns (corresponding to an incident wavelength) summing up to more than 1, thus breaking energy conservation. For the same reason, box filtering the full non-fluorescent component (equation B.1.1) may also lead to a non-energy conserving reradiation matrix.

Mollifying r only. If we only mollify r in the analytical BBRRDF we have no issues. That is because the unmollified $1 - ca$ regulates how much light is available for reflection in the first place. Consider arbitrary $a, r, Q, c \in [0, 1]$ and $e \geq 0$ with $\int e = 1$:

$$\begin{aligned}\int_\Lambda \int_\Omega f(\omega_i, \lambda_i, \omega_o, \lambda_o)\,\mathrm{d}\omega_o\,\mathrm{d}\lambda_o &= \int_\Lambda \mathbb{1}_{|\lambda_i-\lambda_o|<d} \cdot \frac{(1-ca(\lambda_i))r(\lambda_o)}{2d} + ca(\lambda_i)Qe(\lambda_o)\,\mathrm{d}\lambda_o \\ &= \frac{1-ca(\lambda_i)}{2d}\int_{\lambda_i-d}^{\lambda_i+d} r(\lambda_o)\,\mathrm{d}\lambda_o + ca(\lambda_i)Q \\ &\leq \frac{1-ca(\lambda_i)}{2d} \cdot 2d + ca(\lambda_i) = 1.\end{aligned}$$

B.2 Photon Conservation

B.2.1 Symmetric Kernels

If an elastic BBRRDF is mollified with any symmetric kernel $k(\lambda_o|\lambda_i)$ centered around $r(\lambda_i)$, i.e.

$$f_{moll}(\lambda_i, \lambda_o) = r(\lambda_i) \cdot k(\lambda_o|\lambda_i) \cdot \ldots, \tag{B.2.1}$$

the dots representing an optional geometrical part of the BBRRDF which does not depend on wavelengths, and

$$k(\lambda_i + x|\lambda_i) = k(\lambda_i - x|\lambda_i),$$

f_{moll} is not only energy conserving but also photon conserving.

Assume a surface is illuminated with one unit energy at wavelength λ_i. The distribution of spectral irradiance over outgoing wavelengths λ_o is described by the mollified BBRRDF integrated over the hemisphere:

$$E(\lambda_o) = \int_\Omega f_{moll}(\lambda_i, \lambda_o)\,\mathrm{d}\omega_o^\perp = r(\lambda_i) \cdot k(\lambda_o|\lambda_i). \tag{B.2.2}$$

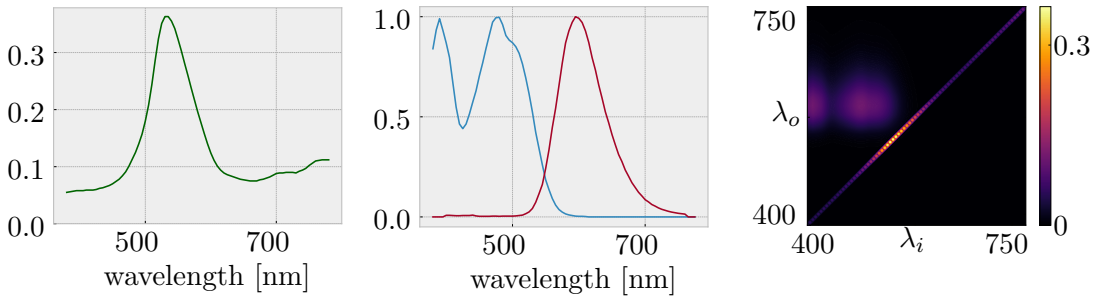


Figure B.1: A reflectance spectrum (**left**) and fluorescence absorption (blue) and emission (red) spectrum (**center**), and the corresponding reradiation matrix (**right**) for the BBRRDF from equation 4.7.1 [8].

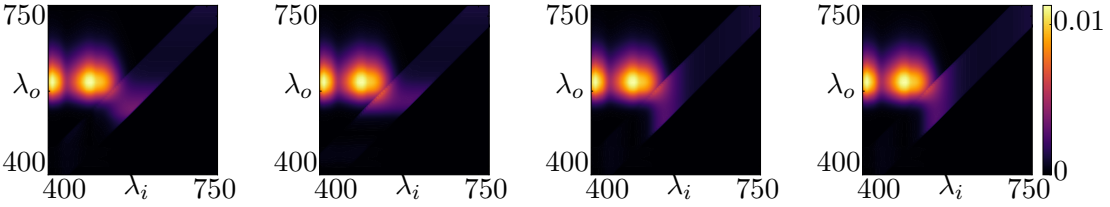


Figure B.2: Based on the reradiation matrix in figure B.1. **Left**: $r(\lambda)$ mollified with f_{mi}. **Center left**: $(1 - ca(\lambda))r(\lambda)$ (i.e. the matrix diagonal) mollified with f_{mi}. The former fulfills energy conservation, but is not directly applicable to the reradiation matrix itself. The latter leaks energy to shorter wavelengths, violating energy conservation. **Center right**: $r(\lambda)$ mollified with f_{mo}. **Right**: $(1 - ca(\lambda))r(\lambda)$, in other words, the matrix diagonal, mollified with f_{mo}. Both are the same and preserve energy conservation.

The energy of a single photon is inversely proportional to its wavelength (equation 4.4.1) We can therefore express the distribution of outgoing photons for one unit of incident energy at λ_i as

$$P_o(\lambda_o) := r(\lambda_i)k(\lambda_o|\lambda_i) \cdot \frac{\lambda_o}{h}.$$

Integrating this yields the total number of outgoing photons per unit incident energy at λ_i:

$$\begin{aligned}
N_{out}(\lambda_i) &= \int_\Lambda r(\lambda_i)k(\lambda_o|\lambda_i)\frac{\lambda_o}{h}\,\mathrm{d}\lambda_o \\
&= \frac{r(\lambda_i)}{h}\int_{-\infty}^{\infty} k(\lambda_i + x|\lambda_i) \cdot (\lambda_i + x)\,\mathrm{d}x \\
&= \frac{r(\lambda_i)}{h}\left(\lambda_i \int_{-\infty}^{\infty} k(\lambda_i + x|\lambda_i)\,\mathrm{d}x + \int_{-\infty}^{\infty} k(\lambda_i + x|\lambda_i)x\,\mathrm{d}x\right) \\
&= \frac{r(\lambda_i)}{h}\left(\lambda_i + \int_{-\infty}^{0} k(\lambda_i + x|\lambda_i)x\,\mathrm{d}x + \int_{0}^{\infty} k(\lambda_i + x|\lambda_i)x\,\mathrm{d}x\right) \\
&= \frac{r(\lambda_i)}{h}\left(\lambda_i + \int_{0}^{\infty} -k(\lambda_i - x|\lambda_i)x\,\mathrm{d}x + \int_{0}^{\infty} k(\lambda_i + x|\lambda_i)x\,\mathrm{d}x\right) = r(\lambda_i)\frac{\lambda_i}{h}.
\end{aligned}$$

On the other hand, the number of photons corresponding to one unit of incident energy at λ_i can be computed according to equation B.2.2 as

$$N_{in} = \frac{\lambda_i}{h}.$$

This means that if $r(\lambda_i) \in [0,1]$, f_{moll} with a symmetric kernel does not produce additional photons, and if $r(\lambda_i) = 1$, the number of outgoing photons is equal to the number of incident photons.

B.2.2 Asymmetric Kernels

Asymmetric kernels do not necessarily result in photon conserving BBRRDFs. For example, the kernel

$$k(\lambda_o|\lambda_i) = \mathbb{1}_{0 \leq \lambda_o - \lambda_i < d}$$

for some mollification distance $d > 0$ only mollifies from shorter to longer wavelengths. While this might be more akin to the typical behavior of fluorescent materials, consider this kernel applied to equation B.2.1. If $r(\lambda_i)$ is large enough, i.e. enough incident energy is reflected, the number of photons required to carry the reflected energy at longer wavelengths is larger than the number of incident photons with wavelength λ_i, since photons with longer wavelengths carry less energy.

B.3 Mollified PDFs

In this section we present the full mollified PDFs for fluorescent BBRRDFs. We start with the model from equation 4.7.1 and provide details on the different mollification options discussed in section 6.3.2.

B.3.1 Constant Mollification

We start with constant mollification, where we always evaluate the reflectance spectrum r for the incident wavelength λ_i. In this case, mollifying r only and mollifying the full non-fluorescent component $(1 - ca)r$ results in the same mollified BBRRDF

$$f_{mo}(\lambda_i \to \lambda_o) = \frac{1}{\pi} \cdot \Big(\frac{\mathbb{1}_{|\lambda_i - \lambda_o| < d} r(\lambda_i)}{2d} \cdot (1 - ca(\lambda_i)) + ca(\lambda_i) Qe(\lambda_o) \Big).$$

The corresponding mollified PDF in camera direction is

$$p_c(\lambda_i|\lambda_o) = \frac{f_{mo}(\lambda_i \to \lambda_o)}{\int_\Lambda f(\lambda \to \lambda_o)\,\mathrm{d}\lambda} \tag{B.3.1}$$

with

$$\int_\Lambda f(\lambda \to \lambda_o)\,\mathrm{d}\lambda = \frac{1}{2d} \int_{\lambda_o - d}^{\lambda_o + d} r(\lambda)(1 - ca(\lambda))\,\mathrm{d}\lambda + cQe(\lambda_o) \int_\Lambda a(\lambda)\,\mathrm{d}\lambda,$$

which can be precomputed.

Similarly, in the direction of a light path the mollified PDF is

$$p_l(\lambda_o|\lambda_i) = \frac{f_{mo}(\lambda_i \to \lambda_o)}{\int_\Lambda f(\lambda_i \to \lambda)\,\mathrm{d}\lambda} \tag{B.3.2}$$

with a slightly simpler

$$\begin{aligned}\int_\Lambda f(\lambda_i \to \lambda)\,\mathrm{d}\lambda &= \frac{1}{2d}\int_{\lambda_o-d}^{\lambda_o+d} r(\lambda_i)(1-ca(\lambda_i))\,\mathrm{d}\lambda + ca(\lambda_i)Q\int_\Lambda e(\lambda)\,\mathrm{d}\lambda \\ &= r(\lambda_i)(1-ca(\lambda_i)) + ca(\lambda_i)Q.\end{aligned}$$

The main issue with this technique is that we need to evaluate a piecewise integral of $(1-ca)r$ (or the diagonal of a reradiation matrix). With piecewise constant spectra this can be done in constant time by precomputing a piecewise linear cumulative distribution function. With reradiation matrices, we can view the diagonal as discrete spectral data, which is simple to integrate. For analytical models, e.g. from spectral upsampling, the integration complexity depends on the model for reflectance and absorption. For instance, the model by [60] can be integrated analytically, but the model by [6] cannot, in which case an approximate solution is required.

Sampling. Note that these PDFs correspond to sampling from the mollified BBRRDF by first sampling whether there is a fluorescent interaction on a camera path

$$\mathbb{P}_c(\lambda_o) = \frac{\int_\Lambda ca(\lambda_i)Qe(\lambda_o)\,\mathrm{d}\lambda_i}{\pi\int_\Lambda f(\lambda_i \to \lambda_o)\,\mathrm{d}\lambda_i}$$

or on a light path

$$\mathbb{P}_l(\lambda_i) = \frac{\int_\Lambda ca(\lambda_i)Qe(\lambda_o)\,\mathrm{d}\lambda_o}{\pi\int_\Lambda f(\lambda_i \to \lambda_o)\,\mathrm{d}\lambda_o},$$

and then sampling the new wavelength, in case of a fluorescent event, from

$$p_c^{fl}(\lambda_i|\lambda_o) = \frac{a(\lambda_i)}{\int_\Lambda a(\lambda)\,\mathrm{d}\lambda}$$

on a camera path and from

$$p_l^{fl}(\lambda_o|\lambda_i) = e(\lambda_o)$$

on a light path, or, in case of a non-fluorescent event, from the mollified non-fluorescent component using

$$p_c^{moll}(\lambda_i|\lambda_o) = \frac{r(\lambda_i)(1-ca(\lambda_i))}{\int_{\lambda_o-d}^{\lambda_o+d} r(\lambda)(1-ca(\lambda))\,\mathrm{d}\lambda}$$

on a camera path and using

$$p_l^{moll}(\lambda_o|\lambda_i) = \frac{\mathbb{1}_{|\lambda_i-\lambda_o|<d}}{2d}$$

on a light path. Combining these yields the full PDF

$$p_x(\lambda_a|\lambda_b) = \mathbb{P}_x(\lambda_b) * p_x^{fl}(\lambda_a|\lambda_b) + (1-\mathbb{P}_x(\lambda_b))p_x^{moll}(\lambda_a|\lambda_b)$$

for $x = c, l$, which equals equations B.3.1 and B.3.2. Also note that for a non-fluorescent BBRRDF this simplifies to $p_x(\lambda_a|\lambda_b) = p_x^{moll}(\lambda_a|\lambda_b)$.

B.3.2 Box Mollification

When mollifying the reflectance component using a 2D box filter, integrating f_{mb} for the PDF evaluation requires evaluating a double integral over the non-fluorescent component $(1-ca)r$. For measured spectra this can still be computed in constant time, based on two precomputed cumulative distribution functions for $(1-ca)r$ and $\int_{-\infty}^{\lambda} r(\lambda)(1-ca(\lambda)\,\mathrm{d}\lambda$.

B.3.3 Reradiation Matrices

For reradiation matrices M with $m_{i,j} = f(\lambda_i \to \lambda_j)$ we get

$$p_c(\lambda_i|\lambda_o) = \frac{m_{i,o}}{\sum_k m_{k,o}}.$$

This means we can get a mollified version of the PDF by using values from the mollified reradiation matrix.

B.4 Additional Results

Figure B.3 illustrate convergence behavior of different sampling strategies. by displaying snapshots at 1, 2, 4, . . . , 64 samples per pixel.

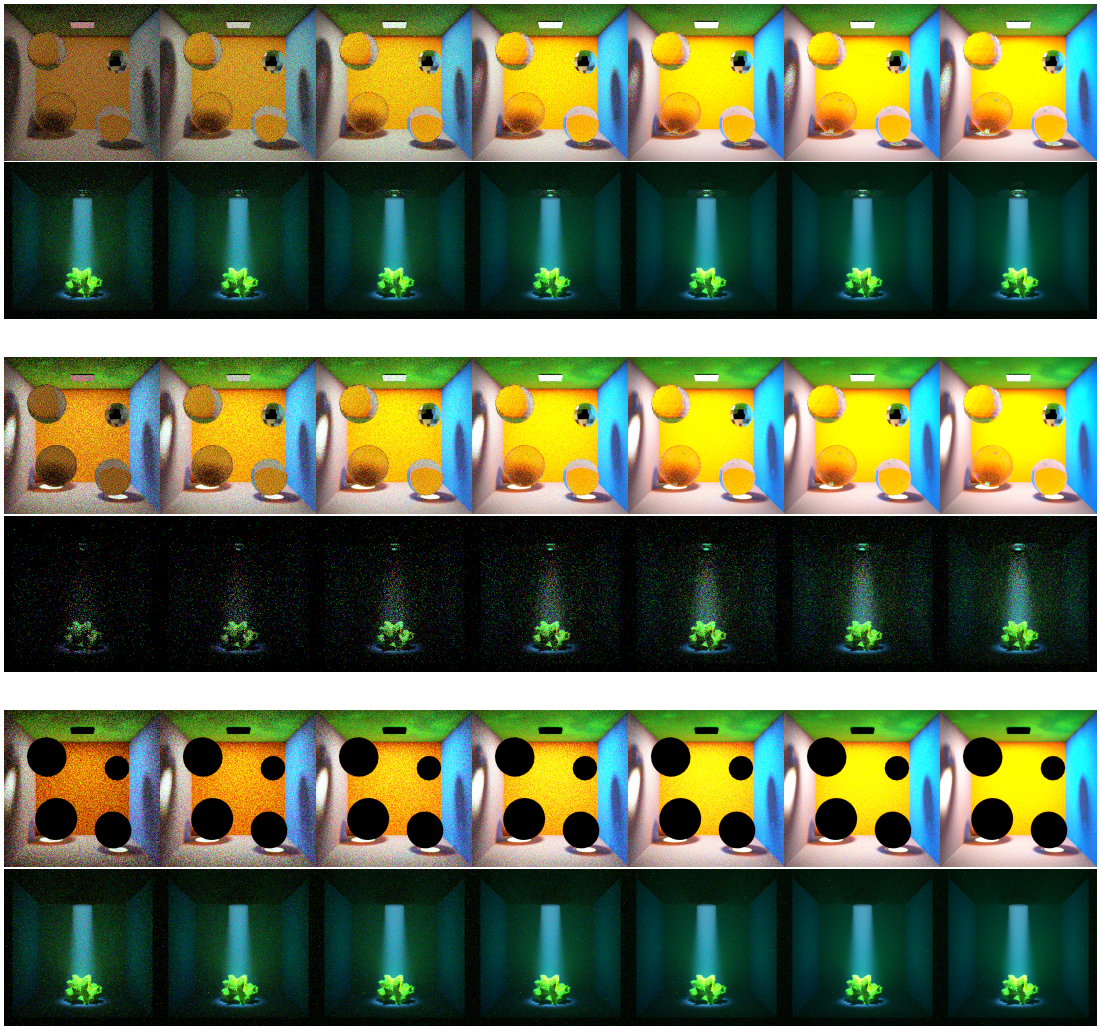


Figure B.3: **Top**: Mollified BDPT, **center**: PT + NEE, **bottom**: LT, in two scenes, at 1, 2, 4, . . . , 64 samples per pixel.

C Appendix: Spatial Mollification for Next Event Estimation with Small Lights

C.1 Specular and Refractive Chains

If there is a longer specular chain $(\mathbf{x}_{d+1}, \dots, \mathbf{x}_s)$ connecting a non-specular vertex $\mathbf{x}_d$ to a point light at $\mathbf{x}_p$ via $(\mathbf{x}_d, \mathbf{x}_{d+1}, \dots, \mathbf{x}_s, \mathbf{x}_p)$, the same reasoning we had behind equation 7.5.10 propagates until we reach $\mathbf{x}_d$, only now we also have to account for distortions of the light source's area due to refractions on top of the accumulated distance:

$$L_o(\mathbf{x}_d \to \mathbf{x}_{d-1}) = I_p f_s(\mathbf{x}_{d+1} \to \mathbf{x}_d \to \mathbf{x}_{d-1}) \prod_{j=d+1}^{s} a_j \cdot \hat{G}(\mathbf{x}_p, \mathbf{x}_d),$$

where $\prod a_j$ accumulates the surface albedos $a_{d+1}, \dots, a_s$, and the Jacobian $\hat{G}$ accounts for the cosines at $\mathbf{x}_d$ and $\mathbf{x}_p$ (with the cosine for a point light being 1), the inverse accumulated distance and potential distortions due to refractions, similar to [145] and the generalized geometry factor in [146]. Figure C.1 provides an intuition for this Jacobian, and section C.1.1 explains why this particular Jacobian arises by modeling point lights as the limit of shrinking disk lights.

Next, we smooth out the emitted light:

$$L_o(\mathbf{x}_d \to \mathbf{x}_{d-1}) \approx \int_{\mathbb{K}_r(\mathbf{x}_p)} \frac{I_p}{A(\mathbb{K}_r)} f_s(\mathbf{x}^*_{d+1}, \mathbf{x}_d, \mathbf{x}_{d-1}) \prod a_j \hat{G}(\mathbf{x}_L, \mathbf{x}_d) \, \mathrm{d}A(\mathbf{x}_L),$$

where $\mathbf{x}^*_{d+1}$ is the position on the first specular surface where $\mathbf{x}_L$ on the kernel plane is visible from $\mathbf{x}_d$. To estimate this with Monte Carlo integration we sample a direction ω_i from the BSDF at $\mathbf{x}_d$ in solid angle density p_σ, ray trace through $s - d$ specular interactions, and intersect with the plane through $\mathbf{x}_p$. Then we convert p_σ to surface area measure at $\mathbf{x}_L$ using the Jacobian $\hat{G}$ so we can use it in the Monte Carlo estimator. In the estimator the Jacobian $\hat{G}$ cancels out, so we never have to actually compute it, leaving us with

$$L_o(\mathbf{x}_d, \mathbf{x}_{d-1}) \approx \begin{cases} \frac{1}{A(\mathbb{K}_r)} \frac{I_p f_s \prod a_j \cdot \cos_{\mathbf{x}_d}(\omega)}{p_\sigma(\omega)} & \text{if } \mathbf{x}_{nee} \in \mathbb{K}_r(\mathbf{x}_p) \\ 0 & \text{else.} \end{cases}$$

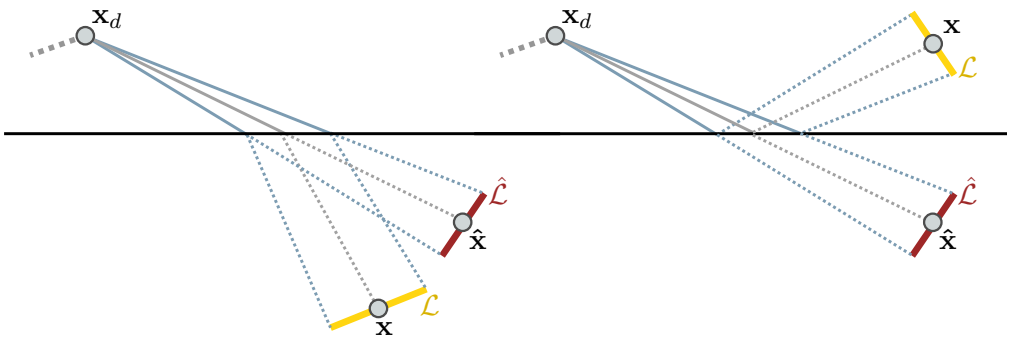


Figure C.1: Visual intuition behind the Jacobian $\hat{G}$. We get the outgoing radiance at $\mathbf{x}_d$ by integrating over the refracted (**left**) or mirrored (**right**) apparent light source $\hat{\mathcal{L}}$ (red) with the usual surface area measure $G(\hat{\mathbf{x}}, \mathbf{x}_d)\,\mathrm{d}A(\hat{\mathbf{x}})$. This is equivalent to integrating over the original light source $\mathcal{L}$ (yellow) with $\hat{G}(\mathbf{x}, \mathbf{x}_d)\,\mathrm{d}A(\mathbf{x})$. The Jacobian $\hat{G} = G \cdot J$ consists of the geometry term between $\mathbf{x}_d$ and $\hat{\mathbf{x}}$ where points $\mathbf{x}$ on $\mathcal{L}$ appear to be, and $J = \mathrm{d}A(\hat{\mathcal{L}})/\,\mathrm{d}A(\mathcal{L})$ to account for the surface distortion due to refraction.

Multiple specular components. If the vertices on the specular chain have multiple specular components, we also sample a discrete component i_j with albedo a_{i_j} for each specular vertex $\mathbf{x}_j$ along the chain with probability $P_j(i_j)$. The rest of the reasoning above stays unaffected, resulting in

$$L_o(\mathbf{x}_d \to \mathbf{x}_{d-1}) \approx \begin{cases} \frac{1}{A(\mathbb{K}_r)} \frac{I_p f_s \prod a_{i_j} \cos_{\mathbf{x}_d}(\omega)}{p_\sigma(\omega) \prod P_j(i_j)} & \text{if } \mathbf{x}_{nee} \in \mathbb{K}_r(\mathbf{x}_p) \\ 0 & \text{else.} \end{cases}$$

C.1.1 Where the Jacobian comes from

In this section we express a point light $\mathbf{x}_p$ with intensity I_p as the limit of a disk area light centered at $\mathbf{x}_p$. The support of the disk light with radius r and normal $\mathbf{n}$ is

$$\mathcal{D}_r = \{\mathbf{x}_p + r \cdot \omega \mid \; ||\omega|| \leq 1 \wedge \langle \omega, \mathbf{n} \rangle = 0\}.$$

We are interested in the outgoing radiance at the last non-specular vertex $\mathbf{x}_d$ before the point light. In order to get the same overall energy at $\mathbf{x}_d$ from the disk as from our target point light, we set the disk's emitted radiance to

$$L_e(\mathbf{x}, \omega) = I_p / A(\mathcal{D}_r)$$

and orient the disk's normal to point in the direction of the mirrored or refracted specular path direction, assuming the disk light is hidden behind a single refraction or reflection (as in figure C.1) with albedo $a(\mathbf{x}_s)$ between $\mathbf{x}_d$ and $\mathbf{x}_p$. We get the outgoing radiance at $\mathbf{x}_d$ by integrating over the surface $\hat{\mathcal{D}}_r$ where the disk light appears to be when viewed from $\mathbf{x}_d$:

$$\begin{aligned} L_o(\mathbf{x}_d \to \mathbf{x}_{d-1}) &= \int_{\hat{\mathcal{D}}_r} L_e(\hat{\mathbf{x}} \to \mathbf{x}_d) a(\mathbf{x}_s) f_s(\hat{\mathbf{x}} \to \mathbf{x_d} \to \mathbf{x}_{d-1}) G(\hat{\mathbf{x}}, \mathbf{x}_d)\,\mathrm{d}A(\hat{\mathbf{x}}) \\ &= \int_{\hat{\mathcal{D}}_r} \frac{I_p}{A(\mathcal{D}_r)} a(\mathbf{x}_s) f_s(\hat{\mathbf{x}} \to \mathbf{x}_d \to \mathbf{x}_{d-1}) G(\hat{\mathbf{x}}, \mathbf{x}_d)\,\mathrm{d}A(\hat{\mathbf{x}}). \end{aligned}$$

In particular, the position $\hat{\mathbf{x}}$ where $\mathbf{x} \in \mathcal{D}_r$ appears to be when viewed from $\mathbf{x}_d$, and the true position $\mathbf{x} \in \mathcal{D}_r$ are the same distance from the mirroring or refractive surface at $\mathbf{x}_s$. In case of refractions, transforming this to an integration over the original disk light requires a Jacobian J for the area distortion between $A(\mathcal{D}_r)$ and $A(\hat{\mathcal{D}}_r)$:

$$L_o(\mathbf{x}_d \to \mathbf{x}_{d-1}) = \int_{\mathcal{D}_r} \frac{I_p}{A(\mathcal{D}_r)} a(\mathbf{x}_s) f_s G(\hat{\mathbf{x}}, \mathbf{x}_d) J(\mathbf{x}, \hat{\mathbf{x}}) \, \mathrm{d}A(\mathbf{x}).$$

If the specular surfaces are flat mirrors, this distortion is zero, i.e. $J \equiv 1$. We can now apply equation 4.8.4 to take this to the limit. We start with a normalized function over a unit disk:

$$f(\mathbf{x}) = \begin{cases} 1/\pi & \text{if } ||\mathbf{x}|| < 1 \\ 0 & \text{else,} \end{cases}$$

with the corresponding function family for disks centered at $\mathbf{x}_p$:

$$f_r(\mathbf{x}) = \frac{1}{r^2} f\left(\frac{\mathbf{x} - \mathbf{x}_p}{r}\right) = \begin{cases} 1/A(\mathcal{D}_r) & \text{if } \mathbf{x} \in \mathcal{D}_r \\ 0 & \text{else.} \end{cases}$$

Next, we bundle the remaining integrand in

$$\phi(\mathbf{x}) = I_p a(\mathbf{x}_s) f_s(\dots) G(\hat{\mathbf{x}}, \mathbf{x}_d) J(\hat{\mathbf{x}}, \mathbf{x}_d),$$

allowing us to write

$$L_o(\mathbf{x}_d \to \mathbf{x}_{d-1}) = \int_{\mathcal{D}_r} f_r(\mathbf{x}) \phi(\mathbf{x}) \, \mathrm{d}A(\mathbf{x}).$$

The integration domain is equal to the support of f_r, so this is equivalent to an integration over an infinite 2D plane. Since the surface area measure A is equivalent to the 2D Lebesgue measure λ^2, equation 4.8.4 now tells us

$$\lim_{r \to 0^+} L_o(\mathbf{x}_d \to \mathbf{x}_{d-1}) = \lim_{r \to 0^+} \int_{\mathcal{D}_r} f_r(\mathbf{x}) \phi(\mathbf{x}) \, \mathrm{d}A(\mathbf{x}) = \phi(\mathbf{x}_p).$$

The cosine at $\mathbf{x}_p$ in the geometry term G is 1 for the direction $\omega^* = \mathbf{x}_p \to \mathbf{x}_s^*$ toward the specular vertex $\mathbf{x}_s^*$ where $\mathbf{x}_p$ is mirrored or refracted, since the disk was placed orthogonal to this direction in the first place, leaving us with

$$\phi(\mathbf{x}_p) = I_p a(\mathbf{x}_s^*) f_s(\mathbf{x}_s^*, \mathbf{x}_d, \mathbf{x}_{d-1}) \frac{\cos_{\mathbf{x}_d} \omega^*}{||\hat{\mathbf{x}}_p, \mathbf{x}_d||} J(\hat{\mathbf{x}}_p, \mathbf{x}_d).$$

We do not see the Jacobian $\hat{G}$ in the final estimator for point lights after specular chains (equation 7.5.11) since we only sample the regularized integral with BSDF samples from a solid angle PDF. Converting that PDF to a surface area PDF for use in the Monte Carlo estimator extends the PDF with that Jacobian, and it cancels out with the Jacobian in the integrand.

C.1.2 Pitfalls with the Dirac Delta Notation

This section highlights why getting the above result using the Dirac delta notation (equation 4.8.3) is not straightforward. Even though the notation allows for point lights or specular surfaces on their own via

$$L_o(\mathbf{x}_s \to \mathbf{x}_d) = \int_{\mathcal{M}} I_p \delta^A_{\mathbf{x}_p}(\mathbf{x}) \cdot f_s(\mathbf{x}, \mathbf{x}_s, \mathbf{x}_d) G(\mathbf{x}, \mathbf{x}_s) \,\mathrm{d}A(\mathbf{x})$$

or

$$L_o(\mathbf{x}_s \to \mathbf{x}_d) = \int_{\Omega} L_i(\mathbf{x}_s, \omega_i) a(\mathbf{x}_s) \delta^\sigma_{\omega_{or}}(\omega_i) \cos_{\mathbf{x}_s} \theta_i \,\mathrm{d}\sigma(\omega_i),$$

we cannot simply use both Dirac deltas in a joint integral. Not only are they tied to different measures, but also the evaluation rule in equation 4.8.3 is only defined if the remaining integrand is a function. If we disregard this and use the rule to resolve the integral for either of the deltas, we get either

$$\begin{aligned} L_o(\mathbf{x}_s \to \mathbf{x}_d) &= \int_{\mathcal{M}} I_p \delta^A_{\mathbf{x}_p}(\mathbf{x}) \cdot a(\mathbf{x}_s) \delta^\sigma_{\omega_{or}}(\omega_i) \cdot G(\mathbf{x}, \mathbf{x}_s) \,\mathrm{d}A(\mathbf{x}) \\ &= I_p a(\mathbf{x}_s) \delta^\sigma_{\omega_{or}}(\omega_p) G(\mathbf{x}_s, \mathbf{x}_p), \text{ or} \\ &\text{or} \\ L_o(\mathbf{x}_s \to \mathbf{x}_d) &= \int_{\Omega} I_p \delta^A_{\mathbf{x}_p}(\mathbf{x}) a(\mathbf{x}_s) \delta^\sigma_{\omega_{or}}(\omega_i) \cos_{\mathbf{x}_s} \theta_i \,\mathrm{d}\sigma(\omega_i) \\ &= I_p \delta^A_{\mathbf{x}_p}(\mathbf{x}^*) a(\mathbf{x}_s) \cos_{\mathbf{x}_s} \theta_i, \end{aligned}$$

with ω_p as the direction from $\mathbf{x}_s$ to the point light, and $\mathbf{x}^*$ as the next visible vertex from $\mathbf{x}_s$ in direction ω_{or}. To fully resolve the remaining Dirac delta we need to use it in the next integral at $\mathbf{x}_d$. At first glance, using the first option with a solid angle integral at $\mathbf{x}_d$ seems promising:

$$\begin{aligned} L_o(\mathbf{x}_d, \omega_o) &= \int_{\Omega} L_i(\mathbf{x}_d, \omega_i) f_s(\omega_i, \mathbf{x}_d, \omega_o) \cos\theta_i \,\mathrm{d}\sigma(\omega_i) \\ &= \int_{\Omega} I_p a(\mathbf{x}_s) \delta^\sigma_{\omega_{or}}(\omega_p) G(\mathbf{x}_s, \omega_o) f_s(\omega_i, \mathbf{x}_d, \omega_o) \cos\theta_i \,\mathrm{d}\sigma(\omega_i) \\ &\overset{?}{=} I_p a(\mathbf{x}_s) G(\mathbf{x}^*_s, \mathbf{x}_p) f_s(\mathbf{x}^*_s, \mathbf{x}_d, \omega_o) \cos\theta_i. \end{aligned}$$

However the last step produces an incorrect result, since the Dirac delta $\delta^\sigma_{\omega_{or}}$ is tied to the solid angle measure at $\mathbf{x}_s$, but the integral uses the solid angle measure at $\mathbf{x}_d$. Doing this accurately involves a measure change for the Dirac delta, which is certainly possible but easy to overlook.

The second option with a surface area integral for $L_o(\mathbf{x}_d \to \mathbf{x}_{d-1})$ runs into a similar problem:

$$\begin{aligned} L_o(\mathbf{x}_d \to \mathbf{x}_{d-1}) &= \int_{\mathcal{M}} L_o(\mathbf{x}_s, \mathbf{x}_d) f_s(\mathbf{x}_s, \mathbf{x}_d, \mathbf{x}_{d-1}) G(\mathbf{x}_s, \mathbf{x}_d) \,\mathrm{d}A(\mathbf{x}_s) \\ &= \int_{\mathcal{M}} I_p \delta^A_{\mathbf{x}_p}(\mathbf{x}^*) a(\mathbf{x}_s) \cos_{\mathbf{x}_s} \theta_i \cdot f_s(\mathbf{x}_s, \mathbf{x}_d, \mathbf{x}_{d-1}) G(\mathbf{x}_s, \mathbf{x}_d) \,\mathrm{d}A(\mathbf{x}_s) \\ &\overset{?}{=} I_p a(\mathbf{x}^*_s) \cos_{\mathbf{x}^*_s}(\mathbf{x}^*_s \to \mathbf{x}_p) f_s(\mathbf{x}^*_s, \mathbf{x}_d, \mathbf{x}_{d-1}) G(\mathbf{x}^*_s, \mathbf{x}_d). \end{aligned}$$

We thus view the Dirac delta as a convenient shorthand notation for configurations with only one singularity in a context where the measure of the singular integral is clear. In the context of measure changes or multiple singularities we think that the notation as a limit of integrals is a valuable alternative that offers more clarity as it explicitly highlights both the measure and necessary Jacobians.

C.2 Environment Maps

In order to apply our method to environment maps following a specular vertex, we approximate the incident radiance at the specular vertex $\mathbf{x}_s$ as

$$L_i(\mathbf{x}_s, \omega_i) = L_{env}(\omega_i) \approx \int_{\mathbb{K}_r(\omega_i)} \mathcal{K}_r(\omega|\omega_i) L_{env}(\omega) \, \mathrm{d}\sigma(\omega) \tag{C.2.1}$$

with a kernel density estimation over directions for a kernel $\mathcal{K}_r(\omega|\omega_i)$ centered at ω_i.

In principle, we can now apply Monte Carlo integration to estimate this integral, by sampling a direction ω_{nee} from the environment map with PDF p_σ^{nee} in solid angle measure:

$$L_i(\mathbf{x}_s, \omega_i) \approx \frac{\mathcal{K}(\omega|\omega_i) L_{env}(\omega)}{p_\sigma^{nee}(\omega)}.$$

The expected value of this estimator is equal to the integral in equation C.2.1. If L_{env} comes from a texture-based or simple analytical environment map, this puts us in a neat situation where we can compute the kernel estimate integral directly by prefiltering L_i. Because of this we chose not to apply our method to environment maps and instead prefilter them on the fly.

Kernel shapes on the unit sphere. Our environment maps are parameterized as

$$\begin{aligned} &[0,1]^2 \to [0,\pi] \times [0,2\pi] \\ &(u,v) \mapsto (\theta = u\pi, \phi = v2\pi). \end{aligned}$$

We construct our kernels directly in texture space. We recognize that this may result in unusual shapes on the sphere of directions and result in effectively smaller kernels the closer we get to the poles.

Isotropic kernel. For the isotropic kernel we first compute a distance d in texture space, either as a global radius r, or as the average distance to the texture coordinates of the directional ray differentials, scaled by r. We then perform trilinear filtering for the mipmap level $\log(d)$ at the texture coordinates of ω_i.

Anisotropic kernel. For the anisotropic kernel we first convert the ray differentials ω_x, ω_y to texture coordinates (u_x, v_x), (u_y, v_y) and take the vectors a_x, a_y from the texture coordinates (u, v) of ω_i to (u_{xy}, v_{xy}).

We scale a_x, a_y by r and construct the axes of an ellipse around (u, v) intersecting (u_{xy}, v_{xy}). In particular, the longer vector of a_x, a_y serves as the major axis, and the minor axis is constructed such that the shorter vector touches the ellipse. We then perform a naive elliptical texture lookup at (u, v) with the two axes.

C.3 Minimizing the Variance of the Combined Estimate

We first minimize the variance for an area light behind a single mirror, and then show that we get the same result for area lights behind specular and refractive chains. The MIS estimator for a light behind a mirror

$$F_{mis} = w_B(\mathbf{x}_L)\frac{L_i f_r \cos}{p_\sigma^B(\mathbf{x}_L)} + w_m(\mathbf{x}_{L,nee})\frac{\mathcal{K}_r L_i f_r \cos}{p_A^{nee}(\mathbf{x}_{nee})p_\sigma^B(\mathbf{x}_L)}$$

has variance

$$\begin{aligned}V(F_{mis}) = &\int_{\mathcal{L}} w_B^2(\mathbf{x}_L)\frac{(L_i f_r \cos)^2}{p_\sigma^B}(\mathbf{x}_L)\,\mathrm{d}A(\mathbf{x}_L)\\ &+ \int_{\mathcal{L}_\cup}\int_{\mathcal{L}_\cap(\mathbf{x}_L)} w_m^2(\mathbf{x}_L,\mathbf{x}_{nee})\frac{(\mathcal{K}_r L_i f_r \cos)^2}{p_A^{nee}p_\sigma^B}(\mathbf{x}_L,\mathbf{x}_{nee})\,\mathrm{d}A(\mathbf{x}_{nee})\,\mathrm{d}A(\mathbf{x}_L)\\ &- \int_{\mathcal{L}} w_B L_i f_r \cos \mathrm{d}A(\mathbf{x}_L)\\ &- \int_{\mathcal{L}_\cup}\int_{\mathcal{L}_\cap(\mathbf{x}_L)} w_m \mathcal{K}_r L_i f_r \cos \mathrm{d}A(\mathbf{x}_{nee})\,\mathrm{d}A(\mathbf{x}_L).\end{aligned}$$

If both estimators were unbiased, the last two terms would combine to $-L_o(x,\omega_o)$ and be irrelevant for the variance, and the variance would be minimized by the balance heuristic. With a biased estimator this is no longer the case, although the difference between L_o and the last two terms becomes negligible with a shrinking kernel and otherwise continuous functions. We thus ignore them and only minimize the first two terms by a pointwise minimization over $\mathbf{x}_L$:

$$w_B = \operatorname{argmin}\ w_B^2\frac{(L_i f_r \cos)^2}{p_\sigma^B} + \int_{\mathcal{L}_\cap(\mathbf{x}_L)}(1-w_B)^2\frac{(\mathcal{K}_r L_i f_r \cos)^2(\mathbf{x}_{nee})}{p_A^{nee}(\mathbf{x}_{nee})p_\sigma^B}\,\mathrm{d}A(\mathbf{x}_{nee}).$$

We ignore f_s, cos and p_σ^B since they are constant in the minimization as we evaluate them for the direction leading up to $\mathbf{x}_L$ (last paragraph in section 7.5.1). We also replace L_i with the emitted radiance L_e at $\mathbf{x}_L$ or $\mathbf{x}_{nee}$. To find the minimum we then set the derivative w.r.t w_B to 0. The derivative of the remaining first term is obviously $2w_B L_e(\mathbf{x}_L)$. For the derivative of the second term we can pull the derivative inside the integral [1] as

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}w_B}\int_{\mathcal{L}_\cap(\mathbf{x}_L)}\ldots &= \int_{\mathcal{L}_\cap(\mathbf{x}_L)}(2w_B-2)\frac{\mathcal{K}_r^2(\mathbf{x}_{nee}|\mathbf{x}_L)L_e^2(\mathbf{x}_{nee})}{p_A^{nee}(\mathbf{x}_{nee})}\,\mathrm{d}A(\mathbf{x}_{nee})\\ &= (2w_B-2)\int_{\mathcal{L}_\cap(\mathbf{x}_l)}\ldots.\end{aligned}$$

Setting the full derivative to zero results in

$$\begin{aligned}&2w_B L_e + (2w_B-2)\int\cdots = 0\\ \Leftrightarrow &2w_B\left(L_e + \int\ldots\right) = 2\int\ldots\\ \Leftrightarrow &w_B = \frac{\int\ldots}{\int\cdots + L_e},\end{aligned}$$

[1] at least in non-pathological cases, i.e. whenever the integrand is a bounded Lebesgue-integrable function of $\mathbf{x}_{nee}$ for each w_B and the derivative f_w exists for all w and $\mathbf{x}_{nee} \in \mathcal{L}_\cap(\mathbf{x}_L)$.

or, without the dots:

$$w_B = \frac{\int_{\mathcal{L}_\cap(\mathbf{x}_L)} \frac{(\mathcal{K}_r L_e)^2}{p_A^{nee}}(\mathbf{x}_{nee})\,\mathrm{d}A(\mathbf{x}_{nee})}{\int_{\mathcal{L}_\cap(\mathbf{x}_L)} \frac{(\mathcal{K}_r L_e)^2}{p_A^{nee}}(\mathbf{x}_{nee})\,\mathrm{d}A(\mathbf{x}_{nee}) + L_e^2(\mathbf{x}_L)}.$$

The integral over $\mathcal{L}_\cap(\mathbf{x}_L)$ makes this equation impractical. We approximate it by restricting $\mathcal{K}_r$ to a constant kernel, and assume that the rest of the integrand does not vary much within $\mathcal{L}_\cap(\mathbf{x}_L)$:

$$w_B \approx \frac{A_{\mathcal{L}_\cap(\mathbf{x}_L)} \frac{(\mathcal{K}_r L_e)^2}{p_A^{nee}}}{A_{\mathcal{L}_\cap(\mathbf{x}_L)} \frac{(\mathcal{K}_r L_e)^2}{p_A^{nee}} + L_e^2} = \frac{A_{\mathcal{L}_\cap(\mathbf{x}_L)} \mathcal{K}_r^2}{A_{\mathcal{L}_\cap(\mathbf{x}_L)} \mathcal{K}_r^2 + p_A^{nee}}.$$

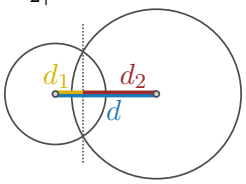
C.3.1 Specular Chains and Refractions

The MIS estimator for regularizing an area light behind longer specular chains and refractions includes the product of specular albedos in the nominators, and the product of selection probabilities for the specular components in the denominator. Assuming that neither varies much over $\mathcal{L}_\cap(\mathbf{x}_L)$, both cancel out in the end, just like the last non-specular BSDF and the PDF $p_\sigma^{BSDF}(\mathbf{x}_L)$ for sampling the direction that will eventually lead to $\mathbf{x}_L$ from that BSDF.
The expected values of the squared BSDF estimator (equation 7.8.1) and squared merge estimator (equation 7.8.3) further include the Jacobian $\hat{G}$ (see appendix C.1). As this Jacobian only depends on the BSDF sample $\mathbf{x}_L$ and not on the NEE sample, it is constant and thus irrelevant for a pointwise minimization over $\mathbf{x}_L$.

C.4 Analytical Overlap of Simple Light and Kernel Shapes

Disk kernels and lights with radii r_1, r_2 and distance d overlap by

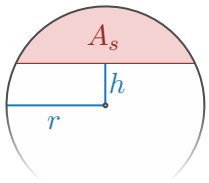
$$A_o^{disk}(r_1, r_2, d) = \begin{cases} 0 & \text{if } r_1 + r_2 < d \\ \pi \min(r_1, r_2)^2 & \text{if } d < |r_1 - r_2| \\ A_s(r_1, d_1) + A_s(r_2, d_2) & \text{else,} \end{cases}$$



with distances d_1, d_2 from the circle center to the intersection plane:

$$d_1 = (r_1^2 - r_2^2 + d^2)/(2d),$$

and $A_s(r, h)$ describes the area of a circle segment with height h from the circle center:



$$A_s(r, h) = r^2 \text{acos}(h/r) - h\sqrt{r^2 - h^2}.$$

Parallel rectangles with radii (r_{*x}, r_{*y}) and distance vector (d_x, d_y) overlap by

$$A_o^{squares}(r_1, r_2, d) = L_o^{lines}(r_{1x}, r_{2x}, d_x) \cdot L_o^{lines}(r_{1y}, r_{2y}, d_y)$$

based on the overlap of 1D intervals

$$L_o^{lines}(r_1, r_2, d) = \begin{cases} 2 \min r_1, r_2 & \text{if } d < |r_1 - r_2| \\ r_1 + r_2 - d & \text{if } d < r_1 + r_2 \\ 0 & \text{else.} \end{cases}$$

Expected overlap of two rectangles. The expected overlap of two aligned rectangles with radii (r_{lx}, r_{ly}) and (r_{kx}, r_{ky}) is based on the expected overlap of their edges

$$\mathbb{E}(A_o^{rectangles}) = \mathbb{E}(L_o^{lines}(r_{lx}, r_{kx})) \cdot \mathbb{E}(L_o^{lines}(r_{ly}, r_{ky})).$$

Since technically the variance for $x_L \notin \mathcal{L}$ is minimal for $w_B = 1$ where the contribution of F_{BSDF} is 0, we evaluate two versions for the expected overlap. First, we consider the expected overlap of two overlapping, axis-aligned rectangles with arbitrary distances

$$\mathbb{E}_{x_L \in \mathcal{L}_\cup}(L_o^{lines}(r_l, r_k)) = 2r_l r_k/(r_l + r_k),$$
$$\mathbb{E}_{x_L \in \mathcal{L}_\cup}(A_o^{rect}) = 4\frac{r_{lx} r_{kx}}{r_{lx} + r_{kx}} \frac{r_{ly} r_{ky}}{r_{ly} + r_{ky}}.$$

Second, we consider the expected overlap if the center x_L of the kernel with radius (r_{kx}, r_{ky}) is restricted to be inside the light source with radius (r_{lx}, r_{ly})

$$\mathbb{E}_{x_L \in \mathcal{L}}(L_o^{lines}(r_{l*}, r_{k*})) = \begin{cases} 2r_{l*} & \text{if } r_{k*} > 2r_{l*} \\ 2r_{l*} r_{k*} - 0.5 r_{k*}^2 & \text{else.} \end{cases}$$

For square lights and kernels with $r_{lx} = r_{ly} =: r_l$ and $r_{kx} = r_{ky} =: r_k$ this simplifies to

$$\mathbb{E}_{x_L \in \mathcal{L}}(A_o^{squares}) = \begin{cases} 4r_l^2 & \text{if } r_k > 2r_l \\ \left(2r_l r_k - 0.5 r_k^2\right)^2 / r_l^2 & \text{else.} \end{cases}$$

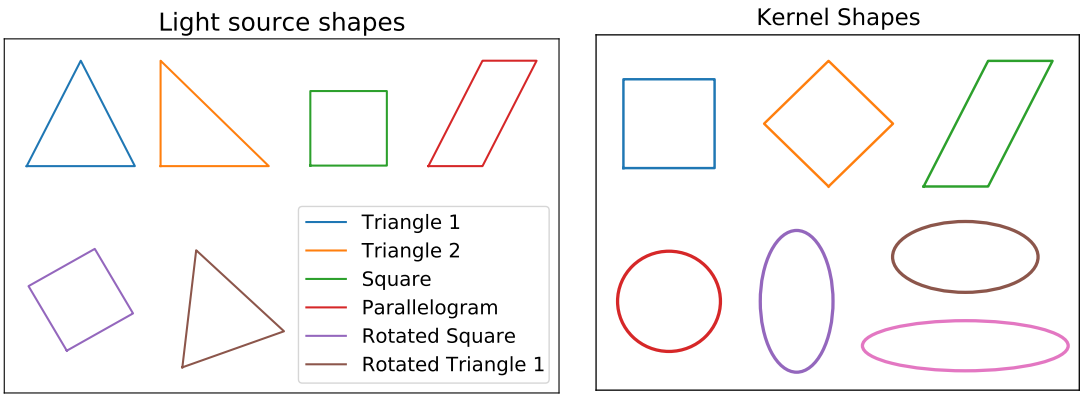


Figure C.2: Light shapes (**left**) and kernel shapes (**right**) used for the average overlap plots in figure C.3.

C.5 Additional Results

C.5.1 Shape Overlap

Computing the overlap between two arbitrary 2D shapes can range from trivial to analytically impossible. Figure C.3 shows the average overlap between typical basic light source and kernel shapes (figure C.2) – triangles and quads for light sources, and circle, parallelogram and ellipse kernels. All overlaps were approximated with brute-force quadrature rule integration, which caused the noise in the data. The kernel shape at scale 1 has the same area as the light source shape. The plots also show the overlap of a unit square (with area one) and scaled unit squares for comparison. The average overlaps are less similar than those of two squares vs. those of two disks (figure 7.10), and the approximation grows worse for more elongated kernels (e.g. the extra wide ellipse in figure C.3 bottom right) and lights.

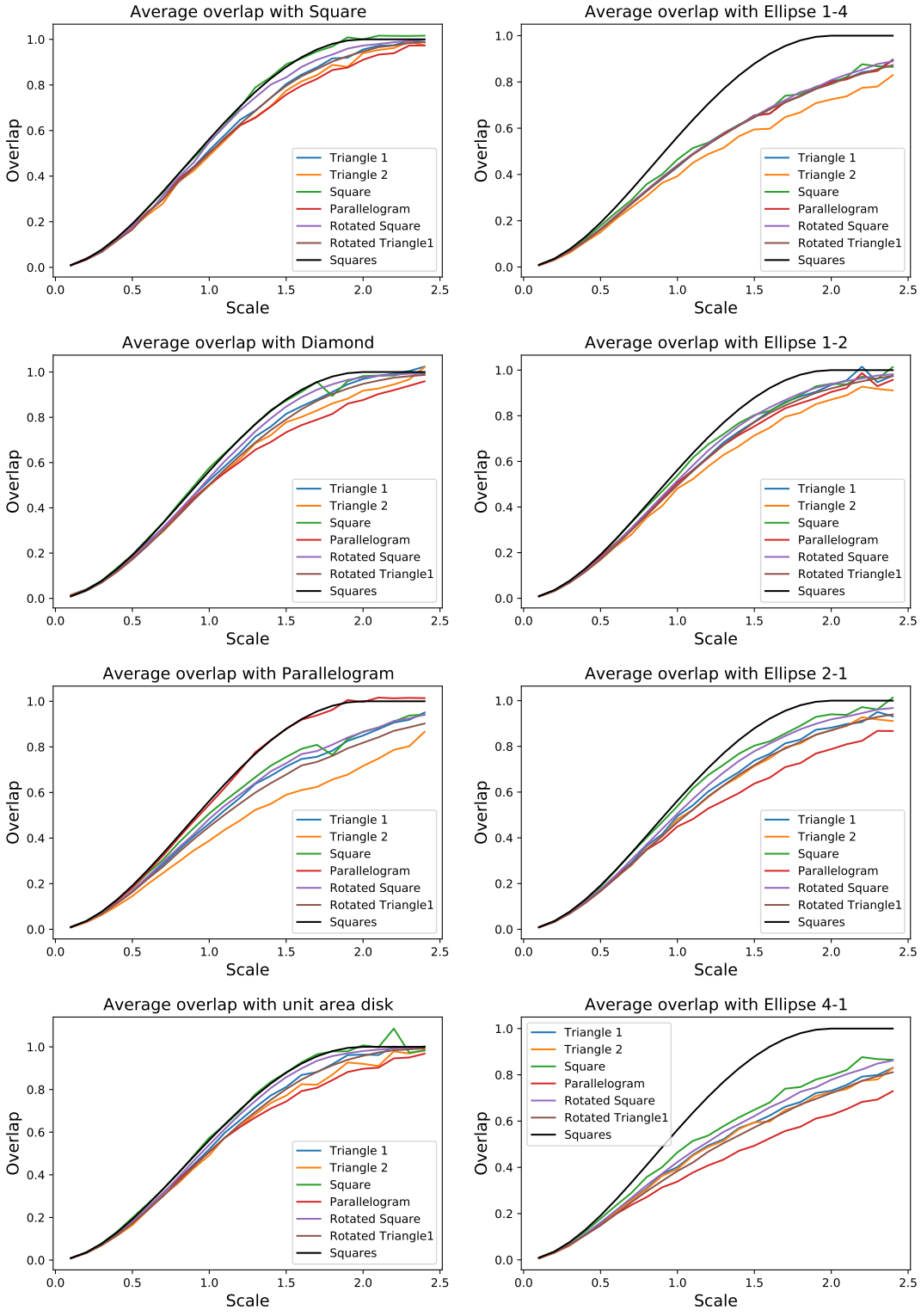


Figure C.3: Overlap between a (kernel) shape with area one, and several (area light) shapes with base area one at different radius scales. The black line shows the average overlap of a unit square with scaled squares for comparison. Kernel shapes **top to bottom**: **Left**: Square, square rotated by 45, parallelogram, circle. **Right**: Extra long ellipse, long ellipse, wide ellipse, extra wide ellipse.

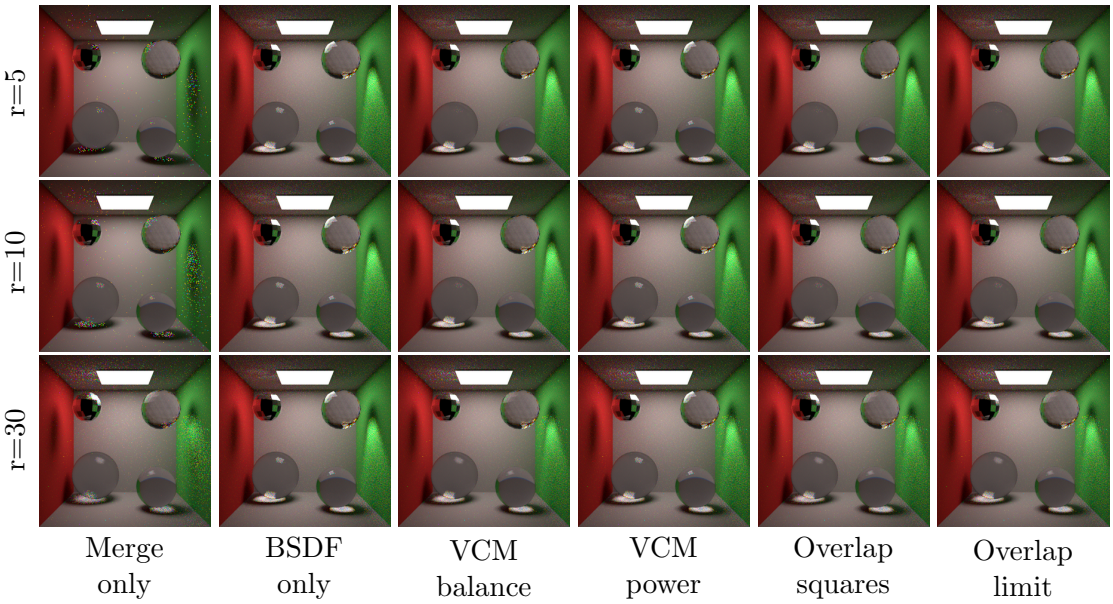


Figure C.4: Large light source, anisotropic kernel scaled by 5 (**top**), 10 (**center**) and 50 (**bottom**) with different MIS weights (**third to last column**) and either only regularization (**first column**) or only BSDF sampling (**second column**).

C.5.2 MIS Weights

Figures C.4, C.5, and C.6 show renders with all kernel shapes for different kernel and light scales. The first column shows regularization only without MIS, followed by BSDF sampling without MIS. Next are the VCM-style balance and power heuristic, followed by the average square overlap heuristic using on the average overlap of two squares with the kernel and light source's area, and the area limit heuristic based on the kernel and light source area. We can observe that all MIS renders turned out quite similar.

Due to file size restrictions the images in figures C.4, C.5 and C.6 are shown at half resolution.

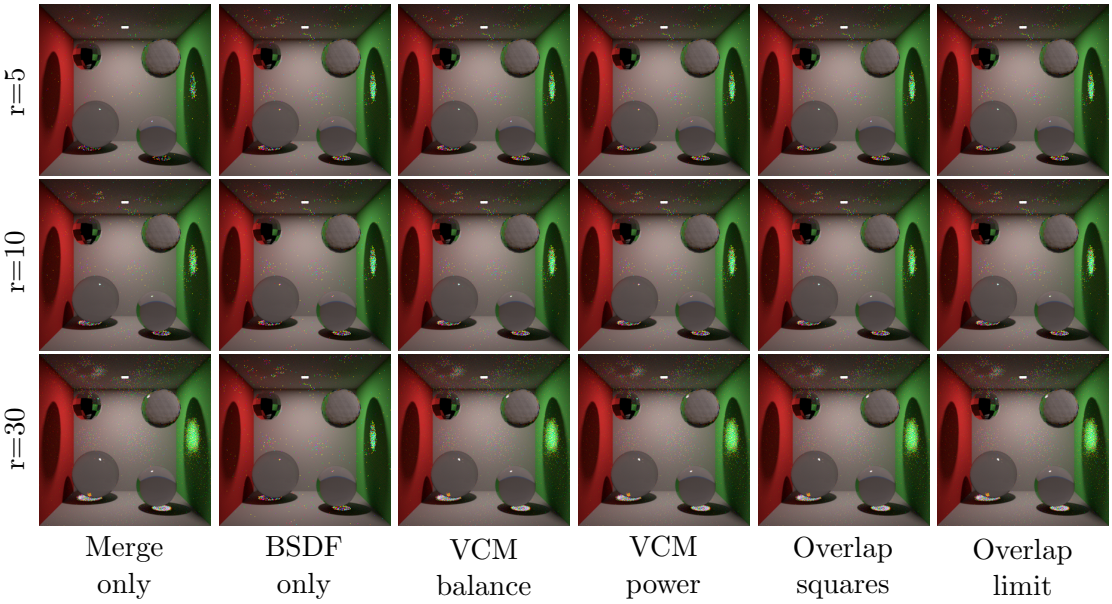


Figure C.5: Medium small light, anisotropic kernel scaled by 5 (**top**), 10 (**center**) and 30 (**bottom**) with different MIS weights (**third to last column**) and either only regularization (**first column**) or only BSDF sampling (**second column**).

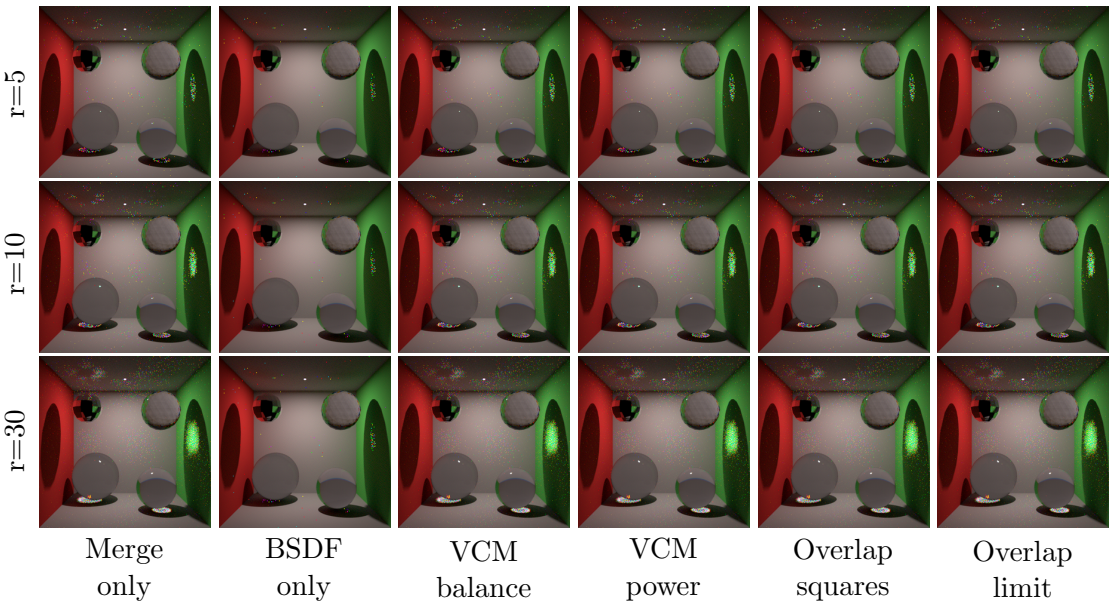


Figure C.6: Small light, anisotropic kernel scaled by 5 (**top**), 10 (**center**) and 30 (**bottom**) with different MIS weights (**third to last column**) and either only regularization (**first column**) or only BSDF sampling (**second column**).

Bibliography

[1] A. Jung, J. Hanika, and C. Dachsbacher, "Spectral Mollification for Bidirectional Fluorescence," *Computer Graphics Forum (Proceedings of Eurographics)*, Jul 2020.

[2] A. Jung and C. Dachsbacher, "Spatial mollification for specular next event estimation," 2022. Unpublished.

[3] A. Jung, A. Wilkie, J. Hanika, W. Jakob, and C. Dachsbacher, "Wide gamut spectral upsampling with fluorescence," in *Computer Graphics Forum*, vol. 38, pp. 87–96, Jul 2019.

[4] L. König, A. Jung, and C. Dachsbacher, "Improving Spectral Upsampling with Fluorescence," in *Workshop on Material Appearance Modeling* (R. Klein and H. Rushmeier, eds.), The Eurographics Association, 2020.

[5] A. Jung, J. Hanika, and C. Dachsbacher, "Detecting Bias in Monte Carlo Renderers using Welch's t-test," *Journal of Computer Graphics Techniques (JCGT)*, vol. 9, pp. 1–25, Jun 2020.

[6] W. Jakob and J. Hanika, "A low-dimensional function space for efficient spectral upsampling," *Computer Graphics Forum (Proceedings of Eurographics)*, vol. 38, Mar 2019.

[7] B. L. Welch, "The generalization of 'student's' problem when several different population variances are involved," *Biometrika*, vol. 34, no. 1/2, pp. 28–35, 1947.

[8] A. Jung, J. Hanika, S. Marschner, and C. Dachsbacher, "A simple diffuse fluorescent bbrrdf model," in *Proceedings of the Eurographics 2018 Workshop on Material Appearance Modeling*, EG MAM '18, (Goslar Germany, Germany), pp. 15–18, Eurographics Association, 2018.

[9] F. Simon, A. Jung, J. Hanika, and C. Dachsbacher, "Selective guided sampling with complete light transport paths," *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, vol. 37, Dec 2018.

[10] V. Schüßler, J. Hanika, A. Jung, and C. Dachsbacher, "Path Guiding with Vertex Triplet Distributions," *Computer Graphics Forum*, vol. 41, Jul 2022.

[11] E. Hamann, A. Jung, and C. Dachsbacher, "Guiding Light Trees for Many-Light Direct Illumination," in *Eurographics 2023 - Short Papers* (V. Babaei and M. Skouras, eds.), The Eurographics Association, 2023.

[12] O. Forster, *Analysis 3 Maß- und Integrationstheorie, Integralsätze im IRn und Anwendungen*. No. 8 in Aufbaukurs Mathematik, Springer Spektrum, 2017.

[13] H.-O. Georgii, *Einführung in die Wahrscheinlichkeitstheorie und Statistik*. Berlin, München, Boston: De Gruyter, 2015.

[14] D. E. Knuth, *The Art of Computer Programming*, vol. 2. Addison Wesley, 3 ed., 1997.

[15] L. Fascione, J. Hanika, D. Heckenberg, C. Kulla, M. Droske, and J. Schwarzhaupt, "Path tracing in production: Part 1: Modern path tracing," in *ACM SIGGRAPH 2019 Courses*, SIGGRAPH '19, (New York, NY, USA), Association for Computing Machinery, 2019.

[16] E. Veach, *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, 1997.

[17] E. Veach and L. J. Guibas, "Optimally combining sampling techniques for monte carlo rendering," in *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, (New York, NY, USA), p. 419–428, Association for Computing Machinery, 1995.

[18] I. Kondapaneni, P. Vévoda, P. Grittmann, T. Skřivan, P. Slusallek, and J. Křivánek, "Optimal multiple importance sampling," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2019)*, vol. 38, pp. 37:1–37:14, Jul 2019.

[19] R. West, I. Georgiev, A. Gruson, and T. Hachisuka, "Continuous multiple importance sampling," *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, vol. 39, Jul 2020.

[20] A. S. Kaplanyan and C. Dachsbacher, "Path space regularization for holistic and robust light transport," *Computer Graphics Forum (Proceedings of Eurographics 2013)*, vol. 32, no. 2, pp. 63–72, 2013.

[21] I. Georgiev, J. Křivánek, T. Davidovič, and P. Slusallek, "Light transport simulation with vertex connection and merging," *ACM Transactions on Graphics*, vol. 31, pp. 192:1–192:10, Nov 2012.

[22] T. Hachisuka, J. Pantaleoni, and H. W. Jensen, "A path space extension for robust light transport simulation," *ACM Transactions on Graphics*, vol. 31, Nov 2012.

[23] B. Walter, S. R. Marschner, H. Li, and K. E. Torrance, "Microfacet models for refraction through rough surfaces," in *Proceedings of the 18th Eurographics Conference on Rendering Techniques*, EGSR'07, (Goslar, DEU), p. 195–206, Eurographics Association, 2007.

[24] M. Pharr, W. Jakob, and G. Humphreys, *Physically Based Rendering: From Theory to Implementation*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 3rd ed., 2016.

[25] W. Jakob, E. d'Eon, O. Jakob, and S. Marschner, "A comprehensive framework for rendering layered materials," *ACM Transactions on Graphics*, vol. 33, Jul 2014.

[26] J. Novák, I. Georgiev, J. Hanika, and W. Jarosz, "Monte Carlo methods for volumetric light transport simulation," *Computer Graphics Forum (Proceedings of Eurographics - State of the Art Reports)*, vol. 37, May 2018.

[27] J. T. Kajiya, "The rendering equation," in *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '86, (New York, NY, USA), p. 143–150, Association for Computing Machinery, 1986.

[28] P. S. Heckbert, "Adaptive radiosity textures for bidirectional ray tracing," in *Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '90, (New York, NY, USA), p. 145–154, Association for Computing Machinery, 1990.

[29] J. Vorba, J. Hanika, S. Herholz, T. Müller, J. Křivánek, and A. Keller, "Path guiding in production," in *ACM SIGGRAPH 2019 Courses*, SIGGRAPH '19, (New York, NY, USA), pp. 18:1–18:77, ACM, 2019.

[30] A. Estevez and C. Kulla, "Importance sampling of many lights with adaptive tree splitting," *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, vol. 1, pp. 1–17, Aug 2018.

[31] P. Moreau and P. Clarberg, *Importance Sampling of Many Lights on the GPU*, pp. 255–283. Berkeley, CA: Apress, 2019.

[32] A. Dittebrandt, J. Hanika, and C. Dachsbacher, "Temporal Sample Reuse for Next Event Estimation and Path Guiding for Real-Time Path Tracing," in *Eurographics Symposium on Rendering - DL-only Track* (C. Dachsbacher and M. Pharr, eds.), The Eurographics Association, 2020.

[33] H. W. Jensen, "Global illumination using photon maps," *Rendering Techniques '96*, pp. 21–30, 1996.

[34] T. Hachisuka, S. Ogaki, and H. W. Jensen, "Progressive photon mapping," *ACM Transactions on Graphics*, vol. 27, Dec 2008.

[35] C. Knaus and M. Zwicker, "Progressive photon mapping: A probabilistic approach," *ACM Transactions on Graphics*, vol. 30, pp. 25:1–25:13, May 2011.

[36] A. S. Kaplanyan and C. Dachsbacher, "Adaptive progressive photon mapping," *ACM Transactions on Graphics*, vol. 32, pp. 16:1–16:13, Apr 2013.

[37] R. Perrot, L. Aveneau, F. Mora, and D. Meneveaux, "Photon mapping with visible kernel domains," *The Visual Computer*, vol. 35, pp. 707 – 720, May 2019.

[38] E. Veach and L. J. Guibas, "Metropolis light transport," p. 65–76, 1997.

[39] C. Kelemen, L. Szirmay-Kalos, G. Antal, and F. Csonka, "A simple and robust mutation strategy for the metropolis light transport algorithm," *Comput. Graph. Forum*, vol. 21, Sep 2002.

[40] L. Belcour and P. Barla, "A Practical Extension to Microfacet Theory for the Modeling of Varying Iridescence," *ACM Transactions on Graphics*, vol. 36, p. 65, July 2017.

[41] S. Werner, Z. Velinov, W. Jakob, and M. Hullin, "Scratch iridescence: Wave-optical rendering of diffractive surface structure," *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, vol. 36, Oct 2017.

[42] L.-Q. Yan, M. Hašan, B. Walter, S. Marschner, and R. Ramamoorthi, "Rendering specular microgeometry with wave optics," *ACM Transactions on Graphics*, vol. 37, pp. 75:1–75:10, Jul 2018.

[43] L. Fascione, J. Hanika, M. Fajardo, P. Christensen, B. Burley, and B. Green, "Path tracing in production – part 1: Writing production renderers," in *SIGGRAPH 2017 Courses*, 2017.

[44] L. Fascione, J. Hanika, M. Leone, M. Droske, J. Schwarzhaupt, T. Davidovič, A. Weidlich, and J. Meng, "Manuka: A batch-shading architecture for spectral path tracing in movie production," *ACM Transactions on Graphics*, vol. 37, pp. 31:1–31:18, Aug 2018.

[45] M. S. Peercy, "Linear color representations for full speed spectral rendering," in *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pp. 191–198, 1993.

[46] G. Ward and E. Eydelberg-Vileshin, "Picture Perfect RGB Rendering Using Spectral Prefiltering and Sharp Color Primaries," in *Eurographics Workshop on Rendering*, pp. 117–124, The Eurographics Association, 2002.

[47] A. Weidlich, A. Forsythe, S. Dyer, T. Mansencal, J. Hanika, A. Wilkie, L. Emrose, and A. Langlands, "Spectral imaging in production: Course notes siggraph 2021," in *ACM SIGGRAPH 2021 Courses*, SIGGRAPH '21, (New York, NY, USA), Association for Computing Machinery, 2021.

[48] "Image courtesy of dr. robert fariss, national eye institute, nih," 2009. Accessed: 2023-06-22, URL: https://medialibrary.nei.nih.gov/media/3505.

[49] "Courtesy of wei li laboratory," 2009. Accessed: 2023-06-22, URL: https://medialibrary.nei.nih.gov/media/3503.

[50] A. Stockman and L. T. Sharpe, "The spectral sensitivities of the middle- and long-wavelength-sensitive cones derived from measurements in observers of known genotype," *Vision Research*, vol. 40, no. 13, pp. 1711–1737, 2000.

[51] "St 2065-1:2021 - smpte standard - academy color encoding specification (aces)," *ST 2065-1:2021*, pp. 1–20, 2021.

[52] I. E. Commission, "Multimedia systems and equipment - colour measurement and management - part 2-1: Colour management - default rgb colour space - srgb : Iec 61966-2-1:1999."

[53] C. I. de l'Eclairage, "Colorimetry – part 4: Cie 1976 l*a*b* colour space : Iso/cie 1164-4," 2019.

[54] J. Meng, F. Simon, J. Hanika, and C. Dachsbacher, "Physically meaningful rendering using tristimulus colours," *Computer Graphics Forum (Proceedings of Eurographics Symposium on Rendering)*, vol. 34, no. 4, pp. 31–40, 2015.

[55] A. Wilkie, S. Nawaz, M. Droske, A. Weidlich, and J. Hanika, "Hero wavelength spectral sampling," in *Proceedings of the 25th Eurographics Symposium on Rendering*, EGSR '14, (Aire-la-Ville, Switzerland, Switzerland), pp. 123–131, Eurographics Association, 2014.

[56] B. Cao, N. Liao, and H. Cheng, "Spectral reflectance reconstruction from rgb images based on weighting smaller color difference group," *Color Research & Application*, vol. 42, no. 3, pp. 327–332, 2017.

[57] D. L. MacAdam, "The theory of the maximum visual efficiency of colored materials," *Journal of the Optical Society of America)*, vol. 25, no. 8, pp. 249–249, 1935.

[58] B. Smits, "An RGB-to-spectrum conversion for reflectances," *Journal of Graphics Tools*, vol. 4, no. 4, pp. 11–22, 1999.

[59] H. Otsu, M. Yamamoto, and T. Hachisuka, "Reproducing spectral reflectances from tristimulus colours," *Computer Graphics Forum*, vol. 37, no. 6, pp. 370–381, 2018.

[60] C. Peters, S. Merzbach, J. Hanika, and C. Dachsbacher, "Using moments to represent bounded signals for spectral rendering," *ACM Transactions on Graphics*, vol. 38, no. 4, 2019.

[61] M. Kerker, "Elastic and inelastic light scattering in flow cytometry," *Cytometry*, vol. 4, Jan 1983.

[62] U. Kubitscheck, *Fluorescence microscopy: from principles to biological applications*. John Wiley & Sons, 2017.

[63] J. Lakowicz, "Principles of fluorescence spectroscopy," *Plenum Press*, vol. 441, Jan 2010.

[64] M. Sauer, J. Hofkens, and J. Enderlein, *Handbook of Fluorescence Spectroscopy and Imaging*. Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim, 2011.

[65] U. Ritgen, *Analytical Chemistry I*. Springer Berlin, Heidelberg, 1 ed., Apr 2023.

[66] H. H. Jaffe and A. L. Miller, "The fates of electronic excitation energy," *Journal of Chemical Education*, vol. 43, p. 469, Sep 1966.

[67] B. Valeur and M. N. Berberan-Santos, *Molecular Fluorescence: Principles and Applications*. John Wiley & Sons, Ltd, 2012.

[68] A. Wilkie, A. Weidlich, C. Larboulette, and W. Purgathofer, "A reflectance model for diffuse fluorescent surfaces," in *Proceedings of the 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia*, GRAPHITE '06, (New York, NY, USA), pp. 321–331, ACM, 2006.

[69] G. U. o. T. Applied Sensor Group, Institute of Analytical Chemistry, "Fluorophores.org database of fluorescent dyes, properties and applications." Accessed: 2023-06-15.

[70] A. S. Glassner, "A model for fluorescence and phosphorescence," in *Photorealistic Rendering Techniques* (G. Sakas, S. Müller, and P. Shirley, eds.), (Berlin, Heidelberg), pp. 60–70, Springer Berlin Heidelberg, 1995.

[71] M. B. Hullin, J. Hanika, B. Ajdin, H.-P. Seidel, J. Kautz, and H. P. A. Lensch, "Acquisition and analysis of bispectral bidirectional reflectance and reradiation distribution functions," *ACM Transactions on Graphics*, vol. 29, pp. 97:1–97:7, Jul 2010.

[72] J. Zwinkels, "Surface fluorescence: the only standardized method of measuring luminescence," *Springer Series on Fluorescence: Standardization and Quality Assurance in Fluorescence Measurements I*, vol. 5, pp. 163–192, 2008.

[73] M. Mojzík, A. Fichet, and A. Wilkie, "Handling fluorescence in a uni-directional spectral path tracer," *Computer Graphics Forum*, vol. 37, no. 4, pp. 77–94, 2018.

[74] Q. Hua, V. Tázlar, A. Fichet, and A. Wilkie, "Efficient storage and importance sampling for fluorescent reflectance," *Computer Graphics Forum*, Nov 2022.

[75] S. Gonzales and M. D. Fairchild, "Evaluation of bispectral spectrophotometry for accurate colorimetry of printing materials," *Proceedings of the Color Imaging Conference*, pp. 14–23, Jun 2000.

[76] S. Gonzales, "Fluorescent database." Accessed: 2022-12-07.

[77] M. Malley, "A heated controversy on cold light," *Archive for History of Exact Sciences*, vol. 42, pp. 173–186, 1991.

[78] E. Becquerel, "Des effets produits sur les corps par les rayons solaires," *Annales de chimie et de physique*, vol. 9, pp. 257–322, 1843.

[79] G. G. Stokes, "Xxx. on the change of refrangibility of light," *Philosophical Transactions of the Royal Society of London*, vol. 142, pp. 463–562, 1852.

[80] F. Stenger, "Zur kenntnis der fluorescenzerscheinungen," *Annalen der Physik*, vol. 28, pp. 201–230, 1886.

[81] O. Lubarsch, "Das stokes'sche gesetz," Jan 1880.

[82] E. Lommel, "Ueber das stokes'sche gesetz," *Annalen der Physik*, vol. 244, pp. 244–253, 1879.

[83] R. Donaldson, "Spectrophotometry of fluorescent pigments," *British Journal of Applied Physics*, vol. 5, pp. 210–214, Jun 1954.

[84] B. Valeur and M. N. Berberan-Santos, *Characteristics of Fluorescence Emission*, ch. 3, pp. 53–74. John Wiley & Sons, Ltd, 2012.

[85] B. Valeur and M. N. Berberan-Santos, "A brief history of fluorescence and phosphorescence before the emergence of quantum theory," *Journal of Chemical Education*, vol. 88, pp. 731–738, Mar 2011.

[86] L.-d. Chiu, T. Ichimura, T. Sekiya, H. Machiyama, T. Watanabe, H. Fujita, T. Ozawa, and K. Fujita, "Protein expression guided chemical profiling of living cells by the simultaneous observation of raman scattering and anti-stokes fluorescence emission," *Scientific Reports*, vol. 7, p. 43569, Mar 2017.

[87] X. Zhu, Q. Su, W. Feng, and F. Li, "Anti-stokes shift luminescent materials for bio-applications," *Chem. Soc. Rev.*, vol. 46, pp. 1025–1039, 2017.

[88] M. Göppert-Mayer, "Über Elementarakte mit zwei Quantensprüngen," *Annalen der Physik*, vol. 401, no. 3, pp. 273–294, 1931.

[89] R. I. Epstein, M. I. Buchwald, B. C. Edwards, T. R. Gosnell, and C. E. Mungan, "Observation of laser-induced fluorescent cooling of a solid," *Nature*, vol. 377, pp. 500–503, 1995.

[90] J. E. Tyler and F. P. Callahan, "On the measurement of "daylight fluorescent" materials," *J. Opt. Soc. Am.*, vol. 41, pp. 997–1001, Dec 1951.

[91] T. F. Scientific, "Fluorescence spectraviewer." Accessed: 2023-06-15.

[92] R. P. Haugland, *Handbook of Fluorescent Probes and Research Chemicals; Molecular Probes Inc.* Molecular Probes, 1996, 6 ed., 1996.

[93] M. Soriano, W. Oblefias, and C. Saloma, "Fluorescence spectrum estimation using multiple color images and minimum negativity constraint," *Opt. Express*, vol. 10, pp. 1458–1464, Dec 2002.

[94] J. Suo, L. Bian, F. Chen, and Q. Dai, "Bispectral coding: compressive and high-quality acquisition of fluorescence and reflectance," *Opt. Express*, vol. 22, pp. 1697–1712, Jan 2014.

[95] M. H. Kim, T. A. Harvey, D. S. Kittle, H. Rushmeier, J. Dorsey, R. O. Prum, and D. J. Brady, "3d imaging spectroscopy for measuring hyperspectral patterns on solid objects," *ACM Transactions on Graphics*, vol. 31, pp. 38:1–38:11, Jul 2012.

[96] Y. Zheng, I. Sato, and Y. Sato, "Spectra estimation of fluorescent and reflective scenes by using ordinary illuminants," in *Computer Vision – ECCV 2014* (D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds.), (Cham), pp. 188–202, Springer International Publishing, 2014.

[97] S. Tominaga, K. Hirai, and T. Horiuchi, "Estimation of fluorescent donaldson matrices using a spectral imaging system," *Opt. Express*, vol. 26, pp. 2132–2148, Jan 2018.

[98] R. D. Hersch, P. Donzé, and S. Chosson, "Color images visible under uv light," in *ACM SIGGRAPH 2007 Papers*, SIGGRAPH '07, (New York, NY, USA), p. 75–es, Association for Computing Machinery, 2007.

[99] D. Gutierrez, F. J. Seron, A. Munoz, and O. Anson, "Visualizing underwater ocean optics," *Computer Graphics Forum*, vol. 27, no. 2, pp. 547–556, 2008.

[100] A. Jarabo and V. Arellano, "Bidirectional rendering of vector light transport," *Computer Graphics Forum*, vol. 37, no. 6, pp. 96–105, 2018.

[101] M. Abdellah, A. Bilgili, S. Eilemann, H. Markram, and F. Schürmann, "A physically plausible model for rendering highly scattering fluorescent participating media," *CoRR*, vol. abs/1706.03024, 2017.

[102] E. Cerezo and F. J. Serón, "Rendering natural waters taking fluorescence into account," *Computer Animation and Virtual Worlds*, vol. 15, no. 3, pp. 471–484, 2004.

[103] A. Wilkie, R. F. Tobler, and W. Purgathofer, "Combined Rendering of Polarization and Fluorescence Effects," in *Eurographics Workshop on Rendering* (S. J. Gortle and K. Myszkowski, eds.), The Eurographics Association, 2001.

[104] D. Gutierrez, A. Munoz, O. Anson, and F. J. Seron, "Non-linear volume photon mapping," in *Proceedings of the Sixteenth Eurographics Conference on Rendering Techniques*, EGSR '05, (Aire-la-Ville, Switzerland, Switzerland), pp. 291–300, Eurographics Association, 2005.

[105] M. Bendig, J. Hanika, H. Dammertz, J. C. Goldschmidt, M. Peters, and M. Weber, "Simulation of fluorescent concentrators," in *Proceedings of IEEE/EG Symposium on Interactive Ray Tracing*, pp. 93–98, 2008.

[106] O. Nalbach, H.-P. Seidel, and T. Ritschel, "Practical capture and reproduction of phosphorescent appearance," *Computer Graphics Forum (Proceedings of Eurographics 2017)*, vol. 36, no. 2, 2017.

[107] J. Jendersie and T. Grosch, "Microfacet model regularization for robust light transport," *Computer Graphics Forum*, vol. 38, pp. 39–47, Jul 2019.

[108] K. O. Friedrichs, "The identity of weak and strong extensions of differential operators," *Transactions of the American Mathematical Society*, vol. 55, pp. 132–151, 01 1944.

[109] D. Dimian and K. Herbst, "Practical uses of a ray tracer for *Cloudy with a Chance of Meatballs*," in *International Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2009, New Orleans, Louisiana, USA, August 3-7, 2009, Talks Proceedings*, 2009.

[110] B. Walter, P. Khungurn, and K. Bala, "Bidirectional lightcuts," *ACM Transactions on Graphics (Processings of SIGGRAPH)*, vol. 31, pp. 59:1–59:11, Jul 2012.

[111] A. Jarabo, J. Marco, A. Muñoz, R. Buisan, W. Jarosz, and D. Gutierrez, "A framework for transient rendering," *ACM Transactions on Graphics (SIGGRAPH Asia 2014)*, vol. 33, no. 6, 2014.

[112] C. I. de l'Eclairage, "Colorimetry – part 6: Cie-de2000 colour-difference formula," 2022. ISO/CIE 11664-6:2022.

[113] A. Celarek, W. Jakob, M. Wimmer, and J. Lehtinen, "Quantifying the error of light transport algorithms," *Computer Graphics Forum*, vol. 38, pp. 111–121, Jul 2019.

[114] "Cornell box comparison." http://www.graphics.cornell.edu/online/box/compare, Cornell University Program of Computer Graphics, Accessed: 2024-01-30.

[115] "The cornell box." http://www.graphics.cornell.edu/online/box/, Cornell University Program of Computer Graphics, Accessed: 2024-01-30.

[116] C. M. Goral, K. E. Torrance, D. P. Greenberg, and B. Battaile, "Modeling the interaction of light between diffuse surfaces," in *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '84, (New York, NY, USA), pp. 213–222, ACM, 1984.

[117] G. W. Meyer, H. E. Rushmeier, M. F. Cohen, D. P. Greenberg, and K. E. Torrance, "An experimental evaluation of computer graphics imagery," *ACM Transactions on Graphics*, vol. 5, pp. 30–50, Jan 1986.

[118] C. Ulbricht, A. Wilkie, and W. Purgathofer, "Verification of physically based rendering algorithms," *Computer Graphics Forum*, vol. 25, no. 2, pp. 237–255, 2006.

[119] F. Drago and K. Myszkowski, "Validation proposal for global illumination and rendering techniques," *Computers & Graphics*, vol. 25, no. 3, pp. 511 – 518, 2001.

[120] R. Mantiuk, K. J. Kim, A. G. Rempel, and W. Heidrich, "Hdr-vdp-2: A calibrated visual metric for visibility and quality predictions in all luminance conditions," *ACM Transactions on Graphics*, vol. 30, pp. 40:1–40:14, Jul 2011.

[121] Z. Wang, A. Bovik, H. Rahim Sheikh, and E. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *Image Processing, IEEE Transactions on*, vol. 13, pp. 600 – 612, May 2004.

[122] G. Wyszecki and W. S. Stiles, *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Wiley classics library, John Wiley & Sons, 2000.

[123] D. L. MacAdam, "Maximum visual efficiency of colored materials," *Journal of the Optical Society of America)*, vol. 25, no. 11, pp. 361–367, 1935.

[124] R. Rossier, *Framework for printing with daylight fluorescent inks*. PhD thesis, École Polytechnique Fédérale de Lausanne, 2013.

[125] S. Agarwal, K. Mierle, and Others, "Ceres solver." http://ceres-solver.org.

[126] M. M. Stark, P. Shirley, and M. Ashikhmin, "Generation of stratified samples for b-spline pixel filtering," *J. Graphics Tools*, vol. 10, no. 1, pp. 39–48, 2005.

[127] ATTO-TEC, "Fluorescence label - atto," 2020. accessed 2020-02-29.

[128] T. Mansencal, M. Mauderer, M. Parsons, N. Shaw, K. Wheatley, S. Cooper, J. D. Vandenberg, L. Canavan, K. Crowson, O. Lev, K. Leinweber, S. Sharma, T. J. Sobotka, D. Moritz, M. Pppp, C. Rane, P. Eswaramoorthy, J. Mertic, B. Pearlstine, and M. Leonhardt, "Colour (python package)."

[129] F. A. Jenkins and H. E. White, *Fundamentals of Optics*. McGraw-Hill, 3 ed., 1957.

[130] S. Popov, R. Ramamoorthi, F. Durand, and G. Drettakis, "Probabilistic connections for bidirectional path tracing," *Computer Graphics Forum*, vol. 34, no. 4, pp. 75–86, 2015.

[131] P. Weier, M. Droske, J. Hanika, A. Weidlich, and J. Vorba, "Optimised path space regularisation," *Computer Graphics Forum*, vol. 40, Jul 2021.

[132] H. Igehy, "Tracing ray differentials," *SIGGRAPH '99*, 1999.

[133] L. Belcour, "A frequency analysis of light transport: from theory to implementation," *PhD dissertation, Université de Grenoble*, 2012.

[134] T. Akenine-Möller, C. Crassin, J. Boksansky, L. Belcour, A. Panteleev, and O. Wright, "Improved shader and texture level of detail using ray cones," *Journal of Computer Graphics Techniques (JCGT)*, vol. 10, pp. 1–24, Jan 2021.

[135] L. Belcour, L.-Q. Yan, R. Ramamoorthi, and D. Nowrouzezahrai, "Antialiasing complex global illumination effects in path-space," *ACM Transactions on Graphics*, vol. 36, Jan 2017.

[136] L. Belcour and C. Soler, "Frequency based kernel estimation for progressive photon mapping," in *SIGGRAPH Asia 2011 Posters*, 2011.

[137] J. Hanika, M. Droske, and L. Fascione, "Manifold next event estimation," *Computer Graphics Forum (Proceedings of Eurographics Symposium on Rendering)*, vol. 34, pp. 87–97, Jun 2015.

[138] G. Loubet, T. Zeltner, N. Holzschuch, and W. Jakob, "Slope-space integrals for specular next event estimation," *ACM Transactions on Graphics*, vol. 39, Nov 2020.

[139] T. Zeltner, I. Georgiev, and W. Jakob, "Specular manifold sampling for rendering high-frequency caustics and glints," *ACM Transactions on Graphics*, vol. 39, Jul 2020.

[140] J. Amanatides, "Ray tracing with cones," *SIGGRAPH*, vol. 18, p. 129–135, Jan 1984.

[141] K. Subr and J. Arvo, "Statistical hypothesis testing for assessing Monte Carlo estimators: Applications to image synthesis," in *15th Pacific Conference on Computer Graphics and Applications (PG'07)*, pp. 106–115, Oct 2007.

[142] F. E. Satterthwaite, "An approximate distribution of estimates of variance components," *Biometrics Bulletin*, vol. 2, no. 6, pp. 110–114, 1946.

[143] B. E. Cooper, "Algorithm as 3: The integral of student's t-distribution," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 17, no. 2, pp. 189–190, 1968.

[144] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables.* New York: Dover, ninth dover printing, tenth gpo printing ed., 1964.

[145] M. Shinya, T. Takahashi, and S. Naito, "Principles and applications of pencil tracing," *SIGGRAPH*, vol. 21, p. 45–54, Aug 1987.

[146] W. Jakob and S. Marschner, "Manifold exploration: A markov chain monte carlo technique for rendering scenes with difficult specular transport," *ACM Transactions on Graphics*, vol. 31, Jul 2012.