Zachodniopomorski Uniwersytet Technologiczny w Szczecinie

Wydział Informatyki

ROZPRAWA DOKTORSKA

# Selected Applications of Perception-based Contrast Models with Varying Adaptation Luminance

(Wybrane obszary zastosowań percepcyjnych modeli przetwarzania kontrastu w warunkach zróżnicowanej adaptacji)

**Mgr inż. Dawid Pająk**
Promotor: Prof. dr hab. inż. Karol Myszkowski

West Pomeranian University of Technology
Computer Science Department
ul. Żołnierska 49
71-210 Szczecin

September 19, 2011

# Acknowledgments

First of all, I would like to express my greatest appreciation to my supervisor, Karol Myszkowski, for his unconditional support, insightful discussions and encouragement throughout the course of this research work.

I am also very grateful to many people with whom I worked during my PhD course: Mohammed Shaheen, Robert Herzog, Martin Čadík, Tunç Ozan Aydın, Elmar Eisemann, Radosław Mantiuk, Robert Strzodka and Makoto Okabe. This dissertation would not have been possible without their help and contributions. I would like to especially thank Radosław Mantiuk for his support, and for convincing me to pursue the PhD degree.

I would also like to thank the dean of our department, prof. Antoni Wiliński, whose help and problem solving skills were invaluable.

I would also like to thank prof. Hans-Peter Seidel for inviting me for an internship with his group at Max-Planck Institute. The experience I gained there defined me as a researcher.

Finally, I am deeply grateful to my family, in particular my mother, Beata, for her constant support and motivation to become a better man.

## Abstract

As human eye is sensitive to contrast, perceptual processing of contrast information is necessary for high-quality, realistic computer graphics. This is even more important nowadays, when real-time processing of visual data is possible with cheap hardware and common display devices allow for high-quality image reproduction. In this thesis we put special focus on contrast processing and show its utility value through several computer graphics applications – from visualization techniques, through multi-scale image editing frameworks, to streaming of rendered synthetic content (depth/geometry-flow). By adding a contrast processing component, we are able to not only perform faithful simulation of selected human visual system aspects, but also improve on the usability of multi-scale image editing frameworks or gain control over the bandwidth allocation in our remote rendering solution. Despite a relatively large computational effort associated with perceptual processing of contrast, modern parallel computing architectures enable a real-time implementation of proposed methods.

## Abstrakt

System wzrokowy człowieka wykazuje większą czułość na kontrast niż na absolutną jasność. Ten fakt jest szczególnie istotny dla grafiki komputerowej, w której duży nacisk kładzie się na realistyczne rezultaty i wysokiej jakości obrazowanie. Percepcyjne modelowanie kontrastu umożliwia przetwarzanie obrazów w sposób zbliżony do człowieka, co z kolei przekłada się na znacznie zwiększony realizm wyświetlanych treści, począwszy od zwykłych zdjęć, poprzez filmy, gry komputerowe czy wizualizacje 3D. W poniższej rozprawie, na przykładzie wybranych aplikacji grafiki komputerowej, pokazujemy użyteczność i skuteczność zaawansowanych modeli postrzegania kontrastu w problemach spotykanych w praktyce. Co ważniejsze, pokazujemy iż modele te znajdują zastosowanie nie tylko w aplikacjach typowych dla obrazowania HDR, jak wizualizacja, ale również w kompresji syntetycznych treści wideo (dane o przepływie optycznym i buforze głębokości) czy edycji obrazów na wielu skalach. Mimo stosunkowo dużych nakładów obliczeniowych związanych z percepcyjnym przetwarzaniem kontrastów, proponowana platforma sprzętowo-programowa oraz zaplecze algorytmiczne umożliwia implementację wspomnianych technik w czasie rzeczywistym.

# Summary

As human eye is sensitive to contrast, embedding contrast perception models into computer graphics seems essential for achieving high-quality imaging and realistic results. The advantages of this approach can be observed in image compression, where contrast metrics focus the encoding process on visually significant information and limit the amount of bits allocated to invisible details, which improves both coding efficiency and final image quality. Such models are also found in HDRI (High Dynamic Range Imaging), where they address, among other things, the reproduction of HDR content on ordinary LDR displays. While beneficial to the above cases, the application of perceptual contrast modeling to computer graphics algorithms is still relatively uncommon. This might have been acceptable in the past, when accurate simulation of our visual system was considered impractical due to its large computational overhead and limited reproduction capabilities of display devices. Nowadays, however, real-time processing of visual data is possible with cheap hardware and common display devices allow for high-quality image reproduction. Consequently, in this dissertation we take a step towards the use of perceived contrast models to improve the visual quality and computational efficiency in important computer graphics applications. Specifically, we consider three problems: interactive contrast manipulation in multi-scale decomposition frameworks, compression of depth and motion video streams, and realistic reproduction of the visibility of photometric images and movies.

## Contrast Prescription for Multi-scale Image Editing

Recently proposed edge-preserving multi-scale image decompositions enable artifact-free and visually appealing contrast enhancement. However, contrast modification in one band usually affects contrasts in other bands, which is not intuitive for the user. In practice, the desired image appearance is achieved through an iterative editing process and often requires fine tuning of contrast in one band several times. To mitigate the above problem and improve the usability of such decompositions, we propose a novel method that utilizes Peli contrast model to restore or "prescribe" the visibility of unmodified sub-band contrast. The proposed contrast prescription method can be incorporated into many existing multi-scale decomposition frameworks, which we demonstrate by extending three state-of-the-art methods. The resulting software implementation, thanks to GPU acceleration, provides a real-time feedback and interaction even for decomposition methods involving PDE solvers.

## Contrast-enhanced Compression of Depth and Motion Video Stream

Contrast information can also be used in, what initially may appear as counter-intuitive, compression of depth and motion data. We demonstrate such a case in a remote rendering system designed for efficient compression and streaming of frames rendered from a dynamic 3D model. Our solution relies on a video augmented by per frame depth and motion information, which is crucial to increasing the robustness of transmission and image reconstruction. Depth and motion share basic characteristics, i.e., they are mostly smooth and need to be precise only at discontinuities (edges), whereas for the remainder of the image we can approximate their values with an edge-preserving diffusion. Based on this key insight we develop a novel depth and motion video encoding scheme that compression-wise outperforms current state-of-the-art encoding standards, such as H.264. The compression method depends on an inherent relation between depth and underlying contrast in rendered images. To capture this relation, we derive a perceptual contrast model that estimates the visual importance of depth edges. By precisely controlling the amount of encoded edges, the proposed method is capable of bandwidth and quality scalability. Our premise is that depth edges are redundant for regions where reconstructed frame contrast is low, as artifacts generated by their absence will most likely be imperceptible. We show that our approach delivers visual quality similar to naïve streaming at a given bandwidth limit and additionally enables client-side applications such as 3D stereo generation or spatio-temporal upsampling.

## Visual Maladaptation in Contrast Domain

To accurately predict the visibility of real-world stimuli, perceptual contrast models account for many non-linear aspects of our visual system. For instance, simplified contrast representations assume global and constant adaptation to display background luminance, which is acceptable for standard 8-bit images and dim displays. However, when considering HDR images or bright displays we need to account for local and temporal nature of adaptation mechanisms. In this work we demonstrate such an approach and concentrate on modeling of maladaptation effect – a local and temporal loss of contrast sensitivity experienced during intense changes in illumination. Although this phenomena is usually fast and does not significantly affect our everyday functioning, its analysis and simulation might be beneficial in vehicle or aircraft control, where accurate prediction of visibility of control equipment would improve the safety and responsiveness in varying illumination conditions. We propose a human visual system model that operates in multi-scale contrast domain and models supra-threshold effects like visual masking, while also accounting for

contrast sensitivity and luminance (mal)adaptation usually considered only in luminance domain frameworks. The time course of adaptation is modeled by considering both neural mechanisms, and pigment bleaching and regeneration. The simulation of maladaptation, as opposed to existing methods, is local, and models the shifting of peak spatial frequency sensitivity in maladapted vision in addition to the uniform decrease in contrast sensitivity among all frequencies. We show the performance of the proposed method in a video processing software, where we visualize maladaptation effects on HDR images and animations at interactive rates.

In summary, this thesis contributes mainly to the fields of multi-scale image editing, video compression and computational models of human visual system. By adding a contrast processing component, we are able to not only perform faithful simulation of selected human visual system aspects, but also improve on the usability of multi-scale image editing frameworks or gain control over the bandwidth allocation in our remote rendering solution. Additionally, we show that despite a relatively large computational effort associated with perceptual processing of contrast, modern parallel computing architectures enable a real-time implementation of proposed methods.

# Streszczenie

System wzrokowy człowieka wykazuje większą czułość na kontrast niż na absolutną jasność. Ten fakt jest szczególnie istotny dla grafiki komputerowej, w której duży nacisk kładzie się na realistyczne rezultaty i wysokiej jakości obrazowanie. Percepcyjne modelowanie kontrastu umożliwia przetwarzanie obrazów w sposób zbliżony do człowieka, co z kolei przekłada się na znacznie zwiększony realizm wyświetlanych treści, począwszy od zwykłych zdjęć, poprzez filmy, gry komputerowe czy wizualizacje 3D. Modele kontrastu oparte na ludzkim systemie wzrokowym są stosowane między innymi w algorytmach kompresji obrazów, gdzie priorytet kodowania informacji jest determinowany przez jej poziom widoczności. Pozwala to na zminimalizowanie artefaktów wynikających ze stratnej kompresji i jednoczesną poprawę postrzeganej jakości obrazu. Percepcja kontrastu jest również wykorzystywana w metodach wizualizacji obrazów HDR, gdzie celem i jednym z głównych problemów jest reprodukcja widocznych kontrastów na standardowych monitorach. Pomimo szerokiego zastosowania w powyższych dziedzinach, percepcyjne modele kontrastu są rzadko spotykane w grafice komputerowej. W przeszłości ich użycie było niepraktyczne z uwagi na dużą złożoność obliczeniową, niską wydajność komputerów oraz ograniczone możliwości wyświetlaczy. Jednak w ostatnich latach sytuacja uległa znacznej poprawie. Przetwarzanie obrazów w czasie rzeczywistym jest osiągalne na ogólnodostępnym sprzęcie, a standardowe monitory komputerowe są w stanie wyświetlać obrazy o wysokiej jakości. W konsekwencji, możliwe jest praktyczne wykorzystanie zaawansowanego modelowania kontrastu co może pozytywnie wpłynąć na działanie wielu metod grafiki komputerowej. Poniższa rozprawa jest krokiem w kierunku stosowania percepcyjnych modeli kontrastu w celu poprawy postrzeganej jakości oraz sprawności obliczeniowej w istotnych aplikacjach grafiki komputerowej. W szczególności rozpatrywane są trzy problemy: interaktywna edycja kontrastu w dekompozycjach wielorozdzielczych, kompresja animacji głębokości i przepływu optycznego oraz realistyczna symulacja widoczności obrazów i animacji o fotometrycznej precyzji.

## Przypisanie kontrastu w metodach edycji obrazów na wielu skalach

Dekompozycja obrazów za pomocą niedawno opracowanych metod wielorozdzielczych umożliwia zachowanie ostrych krawędzi nawet dla sygnałów o niskiej częstotliwości. Mimo iż pozwala to na atrakcyjne wizualnie i pozbawione artefaktów uwydatnianie kontrastu, de-

kompozycje tego typu są mało intuicyjne w użytkowaniu ponieważ modyfikacja kontrastu w jednej rozdzielczości (skali) wpływa na kontrast w sąsiednich skalach. W praktyce pożądany wygląd obrazu jest rezultatem wielu iteracji podczas których kontrast w określonej lokalizacji i skali ulega wielokrotnym zmianom. Aby obejść ten problem i zwiększyć użyteczność wspomnianych dekompozycji proponujemy metodę opartą o metrykę kontrastu Peliego która pozwala na odtworzenie lub przypisanie wartości kontrastu w aktualnie niemodyfikowanych skalach. Zaproponowany algorytm może zostać bezpośrednio zintegrowany z większością istniejących dekompozycji, co demonstrujemy na przykładzie trzech wybranych metod reprezentatywnych dla aktualnego stanu wiedzy. Dzięki zastosowaniu programowalnych układów graficznych, wynikowe oprogramowanie pozwala na interaktywną pracę i edycję obrazów nawet w przypadku metod dekompozycji zdefiniowanych jako rozwiązania układu cząstkowych równań różniczkowych.

## Kompresja animacji bufora głębokości i przepływu optycznego w oparciu o metrykę kontrastu

Mniej oczywistym przypadkiem zastosowania percepcyjnych modeli kontrastu jest kompresja danych bufora głębokości i przepływu optycznego. Przypadek ten rozpatrujemy w kontekście systemu zdalnego renderingu zaprojektowanego w celu efektywnej kompresji i transmisji dynamicznych sekwencji trójwymiarowych. Proponowany system wzbogaca strumień wideo o atrybuty głębokości i przepływu optycznego, które są kluczowe w celu zwiększenia niezawodności transmisji i jakości rekonstrukcji obrazu. Oba atrybuty charakteryzują się podobną strukturą klatki – są w większości gładkie i wymagają precyzyjnych wartości na nieciągłościach (krawędziach). To kluczowe spostrzeżenie pozwoliło nam na opracowanie nowatorskiej metody kompresji głębokości i przepływu optycznego, która zachowuje precyzję na krawędziach a wartości pomiędzy nimi przybliża za pomocą dyfuzji. Wynikowy algorytm kompresji jest efektywniejszy od dostępnych rozwiązań, w tym od standardu kodowania wideo H.264. Skuteczność proponowanej metody opiera się na wykorzystaniu zależności pomiędzy krawędziami w buforze głębokości a kontrastem w przysyłanych klatkach. W celu uchwycenia tej relacji opracowany zostaje model kontrastu który określa wizualną istotność krawędzi w buforze głębokości. W przypadku ograniczenia pasma przesyłu pozwala to na minimalizację widocznych artefaktów poprzez selektywne usuwanie krawędzi, zaczynając od tych odznaczających się niskim kontrastem. Poprzez precyzyjne kontrolowanie ilości kodowanych krawędzi, jesteśmy w stanie skalować jakość i zużycie pasma w zależności od aktualnych ograniczeń. W rezultacie, pokazujemy iż nasze podejście oferuje jakość porównywalną do bezpośrednich rozwiązań zdalnego renderingu i dodatkowo pozwala na atrakcyjne aplikacje po stronie klienta: generowanie obrazu 3D czy też zaawansowane odzyskiwanie rozdzielczości (super-resolution).

## Symulacja efektów maladaptacji w domenie kontrastu

Precyzyjne przewidywanie widoczności kontrastu obserwowanego w rzeczywistości wymaga symulacji wielu skomplikowanych aspektów ludzkiego systemu wzrokowego. Przykładowo, uproszczone modele kontrastu zakładają stałą i globalną wartość adaptacji obserwatora. Mimo iż jest to akceptowalna aproksymacja dla 8-bitowych reprezentacji obrazu i standardowych monitorów, w przypadku obrazów i wyświetlaczy HDR precyzyjne określenie postrzeganego kontrastu wymaga symulacji czasowej i lokalnej natury procesów adaptacji. Podejście to demonstrowane jest w poniższej pracy, gdzie dodatkowo skupiamy się na modelowaniu efektów maladaptacji dla kontrastu, a dokładniej lokalnej i tymczasowej utraty czułości na kontrast spowodowanej nagłą zmianą warunków oświetlenia. Fenomen ten jest z natury krótkotrwały, ale jego analiza i wizualizacja może okazać się przydatna przy projektowaniu paneli kontrolnych i interfejsów odpornych na zmienne warunki oświetlenia, co w konsekwencji może prowadzić między innymi do poprawy bezpieczeństwa w pojazdach samochodowych. Proponowany model systemu wzrokowego przetwarza kontrast na wielu skalach, modeluje efekty ponad-progowe takie jak maskowanie czy zmienna czułość na kontrast oraz symuluje procesy (mal)adaptacji które dotychczas rozpatrywane były w metodach przetwarzających tylko luminancję. W przeciwieństwie do poprzednich rozwiązań, symulacja maladaptacji jest lokalna oraz bierze pod uwagę zmianę kształtu i przesunięcia maksimum funkcji czułości kontrastu w domenie częstotliwości. Proponowana metoda zostaje zaimplementowana w oprogramowaniu przetwarzającym sekwencje HDR, gdzie dzięki sprawnej akceleracji sprzętowej wizualizacja odbywa się w czasie rzeczywistym.

Podsumowując, w poniższej rozprawie pokazujemy użyteczność i skuteczność zaawansowanych modeli postrzegania kontrastu w problemach spotykanych w praktyce. Co ważniejsze, modele te znajdują zastosowanie nie tylko w aplikacjach typowych dla obrazowania HDR, jak wizualizacja, ale również w edycji obrazów na wielu skalach i kompresji syntetycznych treści wideo, takich jak dane o przepływie optycznym i buforze głębokości. Ponadto, mimo stosunkowo dużych nakładów obliczeniowych związanych z percepcyjnym przetwarzaniem kontrastów, pokazujemy iż proponowane algorytmy umożliwiają implementację wspomnianych technik w czasie rzeczywistym na ogólnodostępnym sprzęcie komputerowym.

# Contents

# List of Figures

# List of Tables

# Introduction

As human eye is sensitive to contrast, perceptual processing of contrast information is necessary for high-quality, realistic computer graphics. This is even more important nowadays, when real-time processing of visual data is possible with cheap hardware and common display devices allow for high-quality image reproduction. In this thesis we put special focus on contrast processing and show its utility value through several computer graphics applications ranging from visualization techniques to efficient streaming and compression of rendered synthetic content. Each method utilizes perception-based contrast model and shows significant improvement over already existing approaches. In this chapter we motivate our research and formally define the thesis problem and objectives. At the end, we give the reader a short overview of the accomplished contributions and outline the thesis structure.

## 1.1 Motivation

Eyesight is an important human sense that significantly contributes to the perception of the surrounding world. Not only it allows us to detect variations in the perceived light and color, but also to accomplish complex tasks such as viewing movies, photographs, reading books or exploring new environments. This is all possible because our visual system is optimized to distinguish objects from each other or from their background, even if the corresponding difference in brightness or color is small. The human eye is the most sensitive to photometric luminance (or its perceptual counterpart – brightness), but due to various psychophysical limitations of our visual system we estimate the visibility of perceived world using *contrast*, which can be roughly described as a ratio between the luminance difference and the background luminance [Bar99]. A good example of this visual property are stars. Despite they emit the same amount of light all the time and their luminance difference with respect to the sky remains constant, we can see them only during the night. This is because the night time sky is usually dark, hence the contrast between the stars and the sky is large. In a daytime sky, which is approximately a thousand

times brighter, the corresponding contrast is very small and imperceptible for human eye.

Perception of contrast is particularly important for Computer Graphics (CG) – a sub-field of computer science which studies methods for digitally synthesizing and manipulating visual content. Since their early days CG methods have been aiming at generating images with close to real world appearance. In practice, the perceived quality of images produced by these methods depends not only on the algorithmic part, but also on the display and processing hardware capabilities. For instance, the 8-bit representation of image data, which is still common for both digital capture and display devices, sets a hard limit of 256 possible tones that can be recorded, processed and viewed. Even if we consider complementary color information, this tonal range covers just a fraction of tones we are exposed to while observing scenes in nature. To overcome the inherent drawbacks of such representations and improve on the image reproduction quality, computer graphics methods had to exploit the characteristics of human perception. Accurate simulation of human vision involves highly nonlinear models that work in both spatial and frequency domains, and as the complexity of these models was prohibitive to early computers, the initial vision models were often simplified and constrained to specific viewing conditions. For example, one convenient contrast definition, which is still widely used in image processing, is expressed as a difference between the intensities of neighboring pixels[1]. While this seems like an acceptable approximation for 8-bit image representation, a precise estimation of apparent contrast requires accounting for the frequency of displayed stimulus, the adaptation state of the observer, or even on the contrast of pixels in the surrounding neighborhood.

The need for accurate contrast modeling has become more apparent recently. Nowadays, mainstream display devices can generate more than $500 cd/m^2$ of luminance and reproduce contrast ratios larger than $1000:1$. Moreover, current computers are equipped with powerful floating-point hardware and process visual information with high accuracy and speeds. Despite these recent advances in image display, capture and computing technology, many computer graphics algorithms still assume 8-bit image representation and rely on simplified contrast models, which can negatively impact the results of image processing and lead to suboptimal reproduction quality on current displays. A straightforward solution to the above problem is to employ more advanced perception models that are tailored for current image display and processing capabilities. In this thesis we pursue this thought, and attempt to take a step towards the use of perceived contrast models to improve the visual quality and computational efficiency in important computer graphics applications. Specifically, we consider the following problems:

- ***Contrast Prescription*** — Recently proposed edge-preserving multi-scale (multi-band or multi-resolution) image decompositions enable artifact-free and visually ap-

[1] Precisely, as a difference of gamma-corrected pixel values, which roughly corresponds to a logarithm of their ratio as gamma curve approximates log space in 0..1 range.

pealing contrast enhancement. However, contrast modification in one band (detail level) usually affects contrasts in other bands, which is not intuitive for the user. In practice, the desired image appearance is achieved through an iterative editing process, which often requires fine tuning of contrast in one band several times. To improve the user experience, we introduce a "contrast prescription" feature, which enables the user to lock the contrast in selected areas and bands, in order to make it less sensitive to contrast manipulations in other bands. The prescription method is based on a perceptual contrast metric that introduces a dependency between band-pass image representations. This mathematical connection is later used to "restore" contrast of affected sub-bands.

- ***Contrast-enhanced Compression of Depth and Motion*** — Here we concentrate on a problem of efficient compression and streaming of frames rendered from a dynamic 3D model. Our solution relies on a video augmented by depth and motion (geometry-flow), which are essential for increasing robustness with respect to data loss and image reconstruction. Depth and motion require precise representation of image discontinuities (edges), and because current video encoding methods fail to efficiently encode such features, we propose a custom encoding scheme that stores edges explicitly. To control the bandwidth allocation without affecting the quality of reconstructed frames we need a method that will selectively include edges based on their visual importance and current bandwidth usage. We attempt to address this issue with a non-linear contrast transducer that captures the relation between the magnitude of an edge and underlying luminance contrast. Our premise is that depth edges are redundant for regions where reconstructed frame contrast is low, as artifacts generated by their absence will most likely be imperceptible.

- ***Visual Maladaptation in Contrast Domain*** — It is consciously experienced by everyone that intense changes in illumination temporally cause a loss in visual sensitivity that is later recovered over a time period. Due to this *maladaptation*, the visibility of some scene regions is reduced. The temporal loss of visibility can often be tolerated in daily life, since a large fraction of sensitivity is recovered in just a few seconds. However, driving a car or piloting an airplane requires fast reflexes and undiverted attention at all times. Accurate prediction of visibility could be helpful in reducing our response times, effectively enhancing our safety. We propose a method for simulation of human eye's maladaptation to visual perception over time. Specifically, we attempt to visualize maladapted vision on a display device. Classic contrast models assume global and constant adaptation to display background luminance, which is acceptable for standard image content and dim displays ($< 100 cd/m^2$), such as CRT or old LCD displays. However, when considering HDR (High Dynamic Range) images or bright displays we need to account for local and temporal nature of luminance adaptation mechanisms.

**Figure 1.1** – LDR vs. HDR image comparison

*A typical natural image has a relatively large range of existing tones and their corresponding intensities. In case of 8-bit representation (left), the dark image parts will be clamped to black, very bright to white, while subtle light changes will be lost as a result of quantization. HDR image (right) uses floating point numbers for luminance and color representation, therefore places no restrictions on the tonal range.* (HDR image courtesy of Piotr Didyk)

The idea of using perceptual contrast models to process digital images is not novel. In image and video compression, for example, perception models are employed to determine the visual sensitivity to contrast at different intensities and spatial frequencies. This knowledge helps them to avoid encoding and storing imperceptible image features, which improves the compression ratios and, consequently, reduces the storage requirements. The application of perception models to the compression seems to be especially important nowadays, when high definition content and high resolution displays are widespread, but the cost of data delivery still remains at a relatively high level.

Another field of research that makes an extensive use of perception knowledge is High Dynamic Range (HDR) imaging. In contrast to the traditional approach, where image representation is dictated by hardware limitations, HDR images store data with hardware independent and photometric precision. Such a high-quality description promises significant improvements in image processing and reproduction quality (see Figure 1.1), but also breaks the compatibility with existing LDR display mediums, such as LCD displays, projectors or even paper. Consequently, many state-of-the-art methods account for advanced perception models to efficiently process and reproduce HDR content on LDR display devices.

Although the contrast perception models can be found in HDR imaging and video compression, their application to computer graphics methods is still relatively uncommon. In

this thesis we argue that since such methods produce images and animations intended for humans observers, we can improve their performance and visual quality by accounting for the human perception of contrast. We demonstrate this approach in solutions to various computer graphics problems and show that the use of perception-based contrast models is not limited to typical HDR techniques, such as visualization, but also can be applied to the compression of synthetic video data (depth buffer, geometry-flow) or multi-scale image enhancement.

## 1.2 Problem Statement

The thesis of this work is based on three major claims, all related to the application of perception-based contrast models. First, the use of such models enables the simulation of visual maladaptaion effect. Second, in multi-scale image decomposition frameworks, perceptual contrast models allow to implement contrast prescription. Third, in compression of depth and motion videos, such models improve encoding performance and bandwidth vs. quality scalability.

We show the validity of our claims by fulfilling a set of objectives. The primary objective of this dissertation is to develop contrast models for:

- Local contrast modification in multi-scale image decomposition frameworks that allows to lock or prescribe contrast in a given band while modifying other bands,
- The compression of "visually" significant edges in depth and motion video encoding,
- Simulation of maladapted vision on a display device.

The secondary objective, which is crucial for the primary objective completion, is to develop an efficient implementation of presented solutions. Otherwise, the contrast processing applications will not be responsive enough for a practical use. We reduce this task to a problem of developing an efficient HDR image processing framework that utilizes parallel hardware architectures such as multi-core CPUs and GPUs. Although perceptual contrasts processing is usually associated with large computational effort, we show that, by utilizing dedicated numerical methods and algorithms tailored for the latest parallel hardware architectures, a real-time implementation is possible.

## 1.3 Main Contributions

This section provides a list of publications that constitute the foundation of this dissertation. The fundamental parts of the thesis have been published in international journals and presented at conferences. In particular,

- Chapter 3 is based on an article published in *The Visual Computer Journal* and presented at *Computer Graphics International 2010* conference [PvA*10b],
- Chapter 4 is based on a journal article published in *Computer Graphics Forum* and presented at *Eurographics 2011* conference [PHE*11],
- Chapter 5 is based on a work presented at *IS&T/SPIE's Human Vision and Electronic Imaging 2010* conference [PvA*10a].

This thesis combines these publications under a common context of perceptual contrast model application while presenting improvements and updated results. While the majority of the work presented here is focused around specific functions of perceptual contrast models [PvA*10a, PvA*10b, PHE*11], some significant technical contributions have been also made [MP08, SSPS09, SSPS11, SSPS10, SSP11a, SSP11b] in order to enable efficient implementation of presented applications. The investigation of aforementioned problems resulted in the following key contributions:

- ***A method for per-band contrast restoration and reduction of inter-band energy leakage*** [PvA*10b] — An initial analysis of multi-scale image decomposition frameworks revealed that a modification of a single contrast band affects the appearance of neighboring bands through, so called, "energy leakage". Weak band separation leads to an iterative editing process, where previous changes can be altered or even nullified by the current one, which becomes a serious drawback for the user. We addressed this issue by proposing an algorithm that utilizes Peli contrast model to restore the visibility of contrast in unmodified sub-bands. The resulting method produces less decomposition related artifacts, and allows for a more intuitive and controllable contrast manipulation. Our approach has been incorporated into three state-of-the-art multi-scale decomposition frameworks, for which we show its utility in various image editing scenarios.
- ***A contrast-enhanced compression method of depth and motion video sequences*** [PHE*11] — We have determined that depth and motion need to be precise only at discontinuities (edges), whereas for the remainder of the image we can approximate their values with an edge-preserving diffusion. Consequently, we proposed a custom edge coding scheme followed by a fast diffusion algorithm. The proposed diffusion method maps well to the GPU hardware and reconstructs depth and motion

surfaces in a fixed time, regardless of the scene geometry complexity. To efficiently transmit the edge image we have designed a P-frame based encoding scheme with a novel inter frame prediction model. The proposed compression method includes a contrast metric that allows to precisely control the amount of encoded edges, which is particularly important for bandwidth limited applications. By selectively including edges with high visual importance, we achieved small and controllable distortions in reconstructed frames. The performance of our method was evaluated in a remote rendering system, where we demonstrate that high-quality edge representation enables various client-side application, such as spatio-temporal upsampling, warped 3D stereo or temporal upsampling (increasing frame-rate). The final solution is progressive, scalable, and allows to stream augmented high-resolution frames on a standard hardware. When compared to naïve streaming solutions, our system offers reduced service cost by shifting computations to client GPUs, which improves server-side scalability and balances the server/client workload.

- ***A method for simulation of visual maladaptation over time*** [PvA*10a] — We have proposed a human visual system model that operates in multi-scale contrast domain and models supra-threshold effects like visual masking, while also accounting for contrast sensitivity and luminance (mal)adaptation usually considered only in luminance domain frameworks. Furthermore, the simulation of maladaptation is local and models the shifting of peak frequency sensitivity in maladapted vision, which has not been considered by previous models. The proposed method has been incorporated into image/video processing software, where we visualize maladaptation effects on HDR images and animations at interactive rates. Additionally, we have proposed an extension to existing tone-mapping algorithms that allows for simulation of maladaptation in tone-mapped images.

- ***A computational framework for efficient HDR image processing in heterogeneous hardware environments*** [MP08, SSPS09, SSPS11, SSPS10, SSP11a, SSP11b] — We have developed a collection low-level image processing operations such as vectorized and parallelized gamma-correction, color-space conversion, blending and tone-mapping operators. In addition to this fixed functionality pipeline, we have implemented on top of it a set of more complex image processing algorithms, such as Poisson solver [MMS06], 2nd generation wavelets [Fat09] or multi-scale/scale-space decompositions [BA83, MH80]. To efficiently utilize available computational resources, our framework employs different optimization strategies for GPUs and multi-core CPUs.

## 1.4 Thesis Outline

This dissertation is structured as follows. In the next chapter, we outline the fundamentals of human visual perception, image processing and image/video compression basics relevant to the topics discussed herein. The main contributions of the thesis are divided into three chapters.

In Chapter 3 we present a "contrast prescription" method that enhances multi-scale contrast editing frameworks. Through the use of contrast metric specific to complex images, we reduce decomposition related artifacts and improve the efficiency of image editing process.

Chapter 4 deals with efficient compression and transmission of real-time rendering content. One of the most important parts in our solution is the bandwidth and quality scalability mechanism. We show how a perception-based contrast model can be used to control the bandwidth and improve the compression rates of synthetic data, such as depth or motion-flow. The final solution is evaluated in a comparison with naïve and existing approaches, followed by a demonstration of various applications in a context of a remote rendering system.

In Chapter 5, we further explore the limitations of human vision in order to realistically predict the visibility of HDR scenes. Specifically, we focus on temporal and spatial aspects of retinal adaptation mechanisms that affect contrast perception of depicted scenes. As shown in the results section, the proposed method works for both static images and HDR movie sequences.

In Chapter 6, we conclude the thesis and give an outlook for future work.

Appendix A provides a short overview of technical contributions included in this thesis. We show how parallel and SIMD-capable hardware, such as multi-core CPUs or GPUs, can be used for fast HDR image processing. The proposed methods not only include simple image processing operations, such as gathering or convolution, but also cover acceleration of iterative stencil computations that are commonly used in computational photography applications.

# Background

In this chapter we provide basic terminology and definitions used in the following chapters. We begin with a brief introduction to the human visual system and its characteristics, such as luminance adaptation, contrast sensitivity and visual masking. Then, we present an overview of video compression methods and their relation to human perception to which they owe their high compression efficiency. We summarize this chapter with a short discussion on common and convenient contrast representations and their limitations.

Over the course of evolution, we have developed mechanisms to focus on perceptually "important" signals. As a result, we perceive the surrounding world with great detail, even though the lighting conditions are often less than optimal. These mechanisms also define the limitations of our vision, which are particularly important for computer graphics applications presented in this thesis. For example, knowledge about the eye's sensitivity to certain spatial patterns can be used in methods that aim to reproduce the visibility of the original scene. As we show in Chapter 5, the HVS model we propose predicts the appearance of scene details, as if the viewer was standing in the actual scene depicted by image data. A slightly different approach is demonstrated by current image and video compression techniques outlined in Section 2.11 of this chapter. When looking at images, even though we are not aware of it, not all scene details are equally well visible. This property is heavily exploited in image compression, where visual information is compressed according to its perceptual significance, such that less visible features are stored using fewer bits. In Chapter 4 we show how similar approach can also be applied to the compression of non-visual information, such as depth or motion-flow data.

Decades of psychophysical and physiological studies resulted in multiple models of human vision, each showing different levels of computational complexity and prediction quality. Yet, the exact way in which our visual system works is still unknown, so all of these models remain valid only in the scenarios they have been derived for. Here we focus on early stages of HVS, namely from retina to primary visual cortex, which have been relatively well understood and comprise good numerical approximations. Additionally, since all contrast models presented in this thesis are luminance based, we have limited our discussion regarding the color perception. For a more comprehensive introduction to

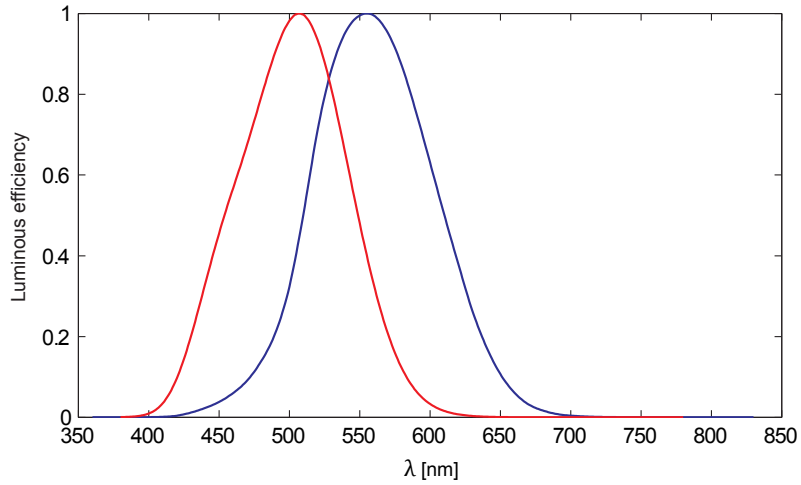


**Figure 2.1** – Luminous efficiency functions

*Photopic (blue) and scotopic (red) luminosity functions. Each function indicates eye's sensitivity to a given wavelength. Based on CIE 1931 standard observer data.* (Data from http://www.cvrl.org/)

color vision, as well as for a more in-depth treatment of the mechanisms discussed in this section we refer the reader to an excellent book by Wyszecki et al. [WS00].

In Chapter 4, we show the use of our novel compression scheme in a remote rendering context. Our proof-of-concept server-side software implements a real-time rendering engine based on a rasterization approach. We refer the reader to series like *Real-Time Rendering* [AMHH08] or *GPU Gems* [PF05] for a thorough introduction to concepts of real-time rendering and deferred shading.

## 2.1 Photometry basics

The principal role of all human vision models is to simulate how we evaluate and process the incoming light sensation. Although we are exposed to a full range of electromagnetic radiation our eyes can register only a small portion of it, called visible light. Since light exhibits wave properties, we can define the visible light spectrum as wavelength interval ranging from about 380*nm* to 760*nm*. There are two standard notations used in measurements of light. First system originates from *radiometry*, which studies the measurement of radiant energy (including light) at all wavelengths in terms of absolute power. However, human eye is not equally sensitive to all wavelengths. To account for that, *photometry* measures the light with wavelength energy weighted with respect to a standardized human perception model. This is realized by scaling the radiant power at each wavelength

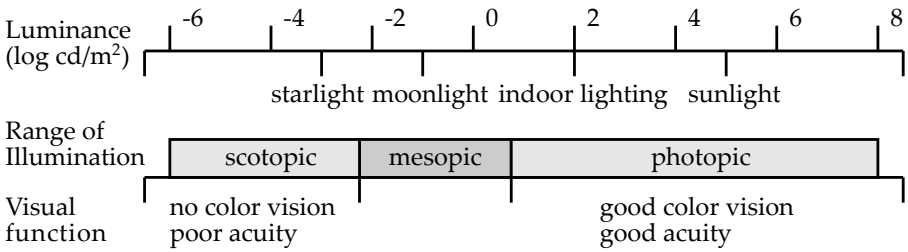


**Figure 2.2** – The range of luminance perceived in the natural environment and corresponding visual properties

*Our visual system demonstrates different characteristics with respect to prevailing lighting conditions in the environment.* (Image redrawn from Ferwerda et al. [FPSG96])

by a *luminosity function* or *luminous efficiency function* $V(\lambda)$ that models eye's brightness sensitivity.

As shown in Figure 2.1, human eye demonstrates different sensitivity values, when adapted to bright (photopic vision) or dark (scotopic) lighting conditions (see Figure 2.2). Typically, photopic sensitivity function is used as a weighting function, which makes photometric measurements inaccurate in dim lighting conditions that we experience, for instance, during the night time. However, in such cases, scotopic sensitivity function can be applied in a similar manner.

In order to describe the light and its perception a basic photometric terminology must be defined. Perhaps, the most important measure of light, from a perspective of computer graphics, is *luminance*, which is often used as an indicator of how bright certain surface (pixel) appears to be. First we introduce the photometric quantities and their corresponding units necessary for deriving and understanding the notion of luminance.

- *Luminous energy* $Q_v$ is the total visible energy (from visible light spectrum) gathered over time from all $n_\lambda$ photons for each wavelengths $\lambda$.

$$Q_v = 683.002 lm/W \cdot \int_0^\infty V(\lambda) n_\lambda e_\lambda \, \mathrm{d}\lambda \quad [lm \cdot s] \tag{2.1}$$

  $e_\lambda$ stands for the energy of a single photon at a given wavelength and is equal to $\frac{h \cdot c}{\lambda}$, where $h \approx 6.63 \cdot 10^{-34} J \cdot s$ is the Planck's constant, and $c$ is the speed of light. Luminous energy is measured in *lumen · second* unit.

- *Luminous flux* or *luminous power* $F$ is a measure of perceived power of a light source.

$$F = \frac{\mathrm{d}Q_v}{\mathrm{d}t} = 683.002 lm/W \cdot \int_0^\infty V(\lambda) J(\lambda) \, \mathrm{d}\lambda \quad [lm], \tag{2.2}$$

  where $J(\lambda)$ is a *spectral power distribution function* and describes how much radi-

ant power is emitted per unit area for a given wavelength $\lambda$. Similarly to luminous energy formulation, the integral incorporates $V(\lambda)$ luminous efficiency function (dimensionless) which reflects human eye sensitivity to particular wavelengths.

- *Illuminance* $E_v$ represents the density of luminous flux incident on a surface and is equal to the total amount of luminous flux per unit area.

$$E_v = \frac{\mathrm{d}F}{\mathrm{d}A} \quad \left[ lux = \frac{lm}{m^2} \right] \tag{2.3}$$

  Analogous formulation stands for *luminous emittance* $M_v$, except it is used to describe the light emitted from a surface.

- *Solid angle* $\Omega$ is the area on a unit sphere cut out by the silhouette of a surface or an object. It is a measure of how large the object appears to an observer placed in the center of the sphere. Small object that are located nearby might subtend the same solid angle as larger objects placed far away. For a sphere with radius $r$ the solid angle can be computed as follows:

$$\Omega = \frac{A}{r^2}, \tag{2.4}$$

  which indicates that the solid angle of an arbitrarily oriented object is obtained by projecting the object onto a sphere and dividing the resulting surface $A$ by the square of sphere radius. From mathematical standpoint, solid angle is a dimensionless quantity, however for convenience it is commonly expressed in *steradians*.

- *Luminance* $L_v$ is a surface orientation dependent (hence the division by the cosine of $\theta$) product of directional density and area density of luminous flux. It can be interpreted as the amount of light energy arriving at surface $dA$ from a given direction with infinitesimal solid angle $d\Omega$. Consequently, it indicates how bright certain surface appears to be.

$$L_v = \frac{\mathrm{d}E_v}{\mathrm{d}\Omega \cos\theta} = \frac{\mathrm{d}^2F}{\mathrm{d}A\mathrm{d}\Omega cos\theta} \quad \left[ \frac{lm}{sr \cdot m^2} = \frac{cd}{m^2} \right], \tag{2.5}$$

  where $\theta$ defines the angle between surface normal and viewing direction. Luminance value is expressed in candelas per square meter units and is often used to characterize light emission from flat diffuse surfaces, such as computer displays.

  An important remark here is that the luminance as a photometric quantity is related but not equivalent to *brightness*, which is a subjective perceptual attribute and as such, cannot be properly measured.

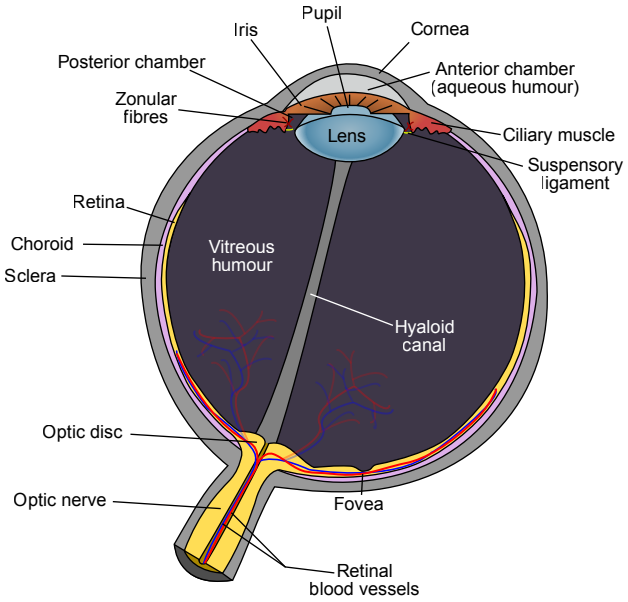


**Figure 2.3** – Optical structure of human eye

*Light passing through pupil is refracted by shape changing lens system and hits retina located on the bottom of an eye. Retina contains two types of photoreceptors: cones and rods. They translate incoming light information into electrical signals, which are then transferred via optic nerve to brain's primary visual cortex for further processing.* (Image from Wikicommons)

## 2.2 Human Eye

Our eye represents an important front-end part of human visual system. Similar to a camera system, it is responsible for registration and initial processing of incoming light information. Figure 2.3 shows the structure of human eye with some key features labeled. Before reaching light-sensitive cells, light passes through an intricate optical system composed of *cornea* and *lens* structures, each having different refraction index. The amount of illumination going through lens is controlled by variable *pupil* size. The final optical image is projected onto the *retina* and through a cascade of complex electrochemical reactions triggers nerve impulses. Retina, the only light-sensitive tissue inside the eye, contains two types of photoreceptor cells: cones and rods. While the primary function of both remains the same (light perception), the performance and ranges of preferred signals are significantly different. Rods are designed to function in dim light conditions (mesopic and scotopic range, see Figure 2.2). They provide relatively low-resolution, monochromatic vision [Fai05], whereas cones, working well under daylight (photopic) illumination, allow for high-resolution vision and color perception. The latter is made possible through three types of cone cells: S (short), M (medium), L (long), designed to distinguish light at different wavelengths.

As shown in Figure 2.4, the distribution of both rods and cones in the retina is non-

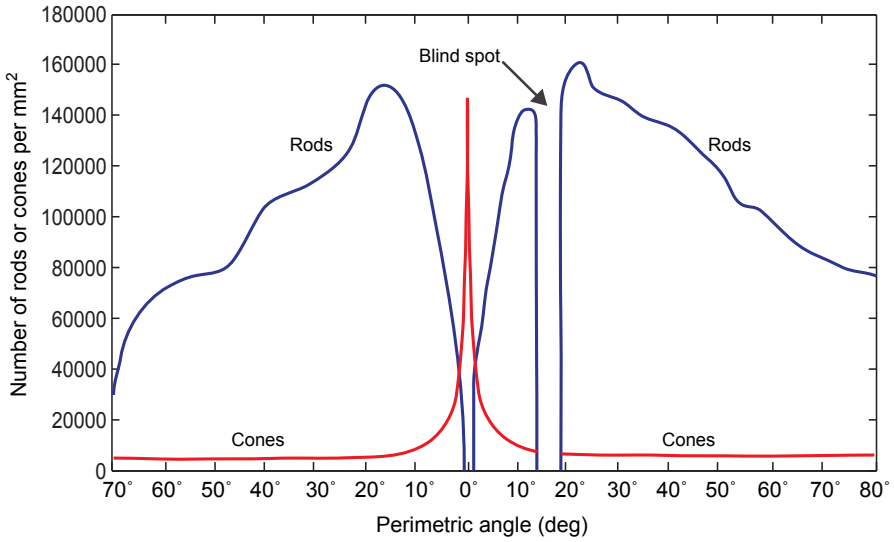


**Figure 2.4** – The distribution of cone and rod cells across the retina

*Cones, most densely packed in the fovea, are responsible for high-resolution daytime vision. Rods, on the other hand, are distributed more uniformly, trading-off spatial resolution for higher light sensitivity.* (Data from [Hec01])

uniform [Hec01]. Cones, gathered near the optic axis – the fovea, enable us to perceive objects on which we focus in high resolution. In contrast, rod cells are spread more evenly, contributing to the perception outside point of gaze. Rods are virtually absent in the fovea, which explains why under dim illumination, we fail to notice low contrast objects (e.g. stars) by looking at them directly, but they become visible once positioned outside the center of gaze.

Interestingly, the fovea comprises less than 1% of the retina size, but close to 50% of optic nerve capacity is allocated for its information. The remaining capacity is used to carry the information from the periphery, which suggests that signals produced by rods in the periphery are significantly more compressed compared to the fovea. This imbalance along with the varying distribution of photoreceptors is one of the reasons of our poor visual acuity in dim lighting.

Signals emitted by rods and cones are processed by retinal ganglion cells. Each ganglion receives input from as few as five photoreceptors in the fovea and up to thousands in the periphery. Together they form a center-surround structure or a *receptive field*, which affects ganglion's response. An important property of the receptive field is that the influence of its center is the opposite of the influence of its surround. About a half of the retinal receptive fields weights positively the center (on-center or excitatory type), while the remainder weights positively the surround region (off-center or inhibitory type). This kind of weighting is mathematically equivalent to edge detection algorithms used in computer graphics. The primary reason of such an encoding is that the raw visual information

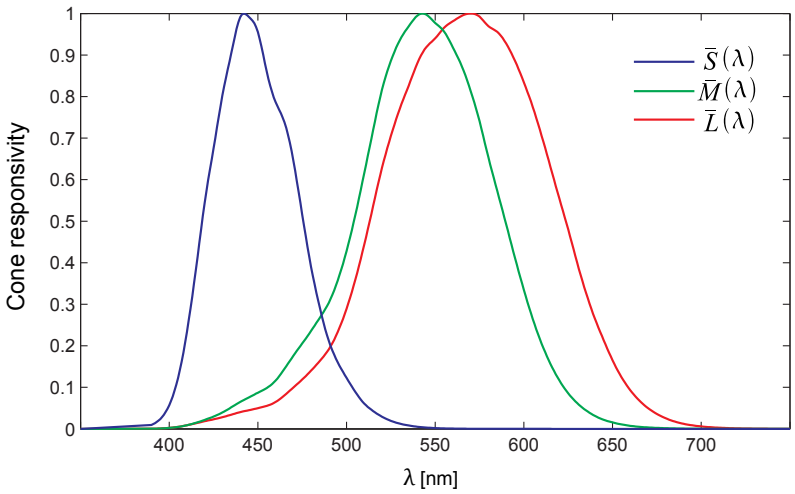


**Figure 2.5** – Spectral sensitivity functions of the S, M, and L cone cells

*Interestingly, the photopic luminous efficiency function $V(\lambda)$ (see Figure 2.1) is a weighted sum of spectral sensitivities of three types of cones. The scaling factor of each cone type is computed according to its relative population size in the retina.* (Data from http://www.cvrl.org/)

produced by rods and cones is too large to be directly transmitted to the brain. The retina solves this issue by employing ganglion cells to spatially encode or "compress" the image data. As a result, the responses from about 125 million of photoreceptors are encoded by 1.2 million ganglion cells and leave the retina through their axons [JSMB*92]. The axons of ganglion cells form the optic nerve at the optic disc, otherwise known as the *blind spot*, due to the lack of rods and cones in the area. Through the optic nerve, the visual information travels to the lateral geniculate nucleus (LGN) and primary visual cortex, where "higher-level" visual processing and analysis takes place.

In computer graphics, the aspect of photoreceptors spatial distribution is often ignored, since gaze independent results are desirable. Therefore, most computational models of human vision assume uniform photoreceptor distribution and combine foveal and peripheral vision into a single and simplified model of the retina.

## 2.3 Colors

In nature, light is a continuous spectrum of visible wavelengths characterized by a spectral power distribution function $J(\lambda)$. As can be seen in Figure 2.5, our visual system has a limited access to this spectrum through three types of cone photoreceptors, with sensitivity peaks in short, middle and long wavelengths. Exposure to light stimuli causes varying excitation of these cones which our brain translates into the sensation of color. The final

responses for light defined by $J(\lambda)$ can be can be computed as follows:

$$S_r = \int_0^\infty \bar{S}(\lambda)J(\lambda)\,\mathrm{d}\lambda, \quad M_r = \int_0^\infty \bar{M}(\lambda)J(\lambda)\,\mathrm{d}\lambda, \quad L_r = \int_0^\infty \bar{L}(\lambda)J(\lambda)\,\mathrm{d}\lambda \tag{2.6}$$

An important consequence of an integral in above formulas is that the color does not have a unique relation in physical world. There is an infinite number of spectral power distributions that would be matched to the same apparent color. This property, known as *metamerism*, is problematic in industries where color matching is important, since apparent surface color matches can disappear under different light sources (e.g. halogen or fluorescent). Also, the spectral responsiveness of cones is largely overlapping, which is in contrast to physical imaging systems that, for practical reasons, aim to reproduce colors using separated color primaries. This difference is the fundamental reason why accurate color reproduction is difficult to realize.

Another consequence of the overlap is a correlation between responses of cones, which means that signals produced by one cone carry information about the excitation of the others. In order to improve coding efficiency and minimize the amount of information that needs to be send to the brain, the visual system encodes light using luminance and two opponent color pairs: red-green and blue-yellow. This kind of encoding, known as *opponent color space*, strongly corresponds to the correlation between coordinates of colors that we see in the real world [Wan95]. Opponent color spaces have been widely used in visual models that aim at faithful color processing [PFFG98, Fai05] as well as in methods that benefit from reduced correlation between color channel, such as compression [TM01] or appearance transfer between images [RAGS01].

The luminous efficiency function was established by CIE (*International Commission on Illumination*) in 1924 and since then a significant effort has been put into development of a colorimetry system that would address metamerism issue. Since the spectral sensitivities of cones were unknown at the time, a set of experiments has been conducted in which human subjects had to visually match particular wavelengths (with unit amount of power) using a mixture of three monochromatic colors, known as *color primaries* or *tristimulus*. Figure 2.6 illustrates spectral tristimulus values for the complete visible spectrum. Together they form so called *color matching functions* or *color mixture functions* and are denoted by:

$$R = \int_0^\infty \bar{r}(\lambda)J(\lambda)\,\mathrm{d}\lambda, \quad G = \int_0^\infty \bar{g}(\lambda)J(\lambda)\,\mathrm{d}\lambda, \quad B = \int_0^\infty \bar{b}(\lambda)J(\lambda)\,\mathrm{d}\lambda \tag{2.7}$$

Notice that some values plotted in Figure 2.6(left) are negative. To cover the entire visible range of colors, human observers were allowed to add monochromatic primary light to reference patch (effectively shifting the plot vertically), if mixing primaries alone was

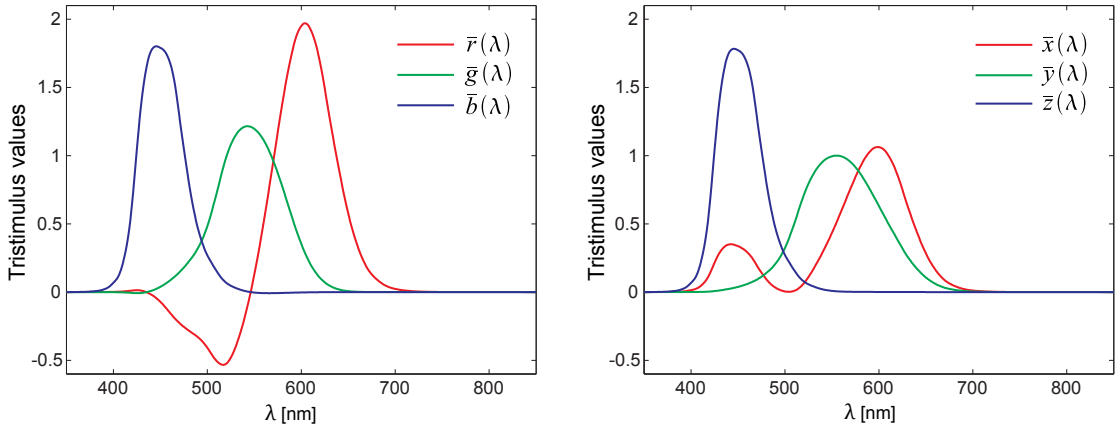


**Figure 2.6** – CIE RGB/XYZ color matching functions

*(left) CIE color matching functions for red (R, 700nm), green (G, 546.1nm) and blue (B, 435.8nm) primaries, (right) CIE XYZ color matching functions. Notice that $\bar{y}$ shape is equal to a photopic luminous efficiency function $V(\lambda)$ shape. Based on CIE 1931 standard observer data.* (Data from http://www.cvrl.org/)

not enough to match apparent colors. In order to eliminate negative tristimulus values and simplify computations a CIE XYZ color matching functions were introduced, Figure 2.6(right). Derived as a linear transformation of CIE RGB, they also cover all visible colors and additionally include photopic luminance efficiency function $V(\lambda)$(denoted as $\bar{y}$). Despite its age, CIE XYZ is still considered a standard in photometric data representation. Linear transformations to and from CIE XYZ are available for all common color spaces [RHD*10].

CIE xyY represents a color space where chromacity coordinates $x = X/(X+Y+Z), y = Y/(X+Y+Z)$ are decorrelated from color brightness $Y$. We can use chromacity coordinates to plot a 2D *chromacity diagram*, which defines the *color gamut*, a complete subset of colors reproducible by a given color space.

In Section 2.11, we show that lossy video encoding schemes use YCbCr color space, which can be thought of as a variant of an opponent color space. In contrast to CIE xyY, limited color gamut and built-in *gamma-compression* (see below) makes this color representation more compact and suitable for common display devices. Such algorithms also exploit our limited perception of color, and apply more aggressive compression to chrominance by means of spatial subsampling and stronger quantization. This is justified by the fact that although we processes brightness and chrominance simultaneously, the relative sensitivity to chrominance is low [Wan95].

## 2.4 HDR and LDR Image Representations

Images in computer graphics are typically defined as rectangular matrices with each entry or *pixel* representing the color information at a discrete location. We can distinguish two types of pixel value representations: LDR (*Low Dynamic Range*) and HDR (*High Dynamic Range*).

LDR pixels describe colors reproduced on a display device and as such they inherit its gamut and range limitations. An example of such a *display-referred* representation is sRGB color space, where each color channel is described by an integer number (e.g., 8-bits per channel) that corresponds to certain display input voltage $V$.

Early computer displays were based on a cathode ray tube (CRT) technology that features a non-linear relation between applied voltage and generated luminance:

$$L_v \approx (L_{peak} - L_{black}) \cdot V^{\gamma} + L_{black}, \tag{2.8}$$

where $V$ is a normalized input voltage, $\gamma \approx 2.4$, $L_{peak}$ and $L_{black}$ are display's peak and minimum light emission respectively. To correctly reproduce colors, prior to sending to the display, each color channel has to be *gamma-corrected*, i.e. transformed by the inverse display model:

$$R_d = R^{1/\gamma}, \quad G_d = G^{1/\gamma}, \quad B_d = B^{1/\gamma} \tag{2.9}$$

Although the CRT nonlinearity allocates more bits to small luminance values, which is consistent with our perception of brightness (see Section 2.8), images encoded in sRGB store color information in a gamma-corrected format, primarily to better utilize the available integer space [Poy98]. The precise gamma-correction formula for sRGB color space is given by:

$$\{R_d, G_d, B_d\} = \begin{cases} 12.92 C_{linear} & , C_{linear} \leq 0.0031308 \\ 1.055 C_{linear}^{1/2.4} & , C_{linear} > 0.0031308 \end{cases}, \tag{2.10}$$

where $C_{linear}$ denotes the linear value of corresponding color channel.

Conversely, HDR images are *scene-referred*, as they describe colors in actual scene. Each HDR pixel uses high-precision integer or floating-point numbers to define tristimulus values (e.g., XYZ color space) describing the complete information about visible light in the scene [RHD*10]. The use of such a high-quality representation is beneficial in computer graphics, where existing image processing methods can be enhanced with human perception models. Since HVS models are derived from experiments conducted using photometric units, such an enhancement is natural and straightforward for HDR images. This in turn translates into significantly increased realism of displayed content, ranging from ordinary pictures, through videos, computer games and visualizations. HDR images (or HDR video

sequences) can also be used in systems based on image processing and analysis, where high image quality and details might have a major impact on the outcome of, for example, face recognition, data extraction from medical records or object tracking in video sequences.

However, as a consequence of unrestricted color representation, displaying HDR images on LDR medium, such as LCD display or paper, requires *tone-mapping*. Tone mapping is a lossy transformation that leads to contrast compression and reduction of image tonal range. A variety of tone mapping operators (TMO) have been developed that approach the problem from different angles. Some try to reproduce high visibility of details in depicted scenes, while others aim to preserve perceived contrast or appearance. The ill-posed inverse operation that converts LDR images back to HDR representation is called inverse TMO [RHD*10].

## 2.5 Luminance Adaptation

The illumination levels we experience vary over many orders of magnitude. As shown in Figure 2.2, our visual system functions over a luminance range of approximately 14 log units, allowing us to see in very dark and bright lighting conditions. However, neural units that compose the system have biophysical limitations that effectively limit their response range to only about 1.5 log units. The visual system solves this disparity between the range of ambient illumination and response capabilities of neurons by employing adaptation mechanisms. Adaptation to a given background luminance enables us to process full range of ambient illumination, although our sensitivity to light spans over 2–3 orders of magnitude around the current adaptation level. In consequence, we are not able to see stars during the day, as we are adopted to daylight illumination and sensitivity to such a dim source is very low. Adaptation mechanisms are particularly important for human vision, as without these mechanisms, our ability to detect and discriminate visual information would be greatly reduced.

While the adaptation is realized by the combined action of pupil and retina, the neural and photochemical processes in the latter are responsible for the majority of the adaptation task. The amount of light received by the retina is proportional to the diameter of pupil, however its variation can change the retinal illuminance only by an order of magnitude, so pupillary action alone is not sufficient to achieve visual adaptation over the full range of light intensities. Actually, it is believed that the main purpose of variation in pupil size is to reduce the optical aberration of eye's lens. Exposed to bright light, the pupil decreases its diameter, effectively reducing the aberration effects, so we can perceive high-resolution images. Consequently, under dim lighting, pupil opens up to provide more light to the retina while reducing the quality of retinal image. The retinal adaptation mechanism is

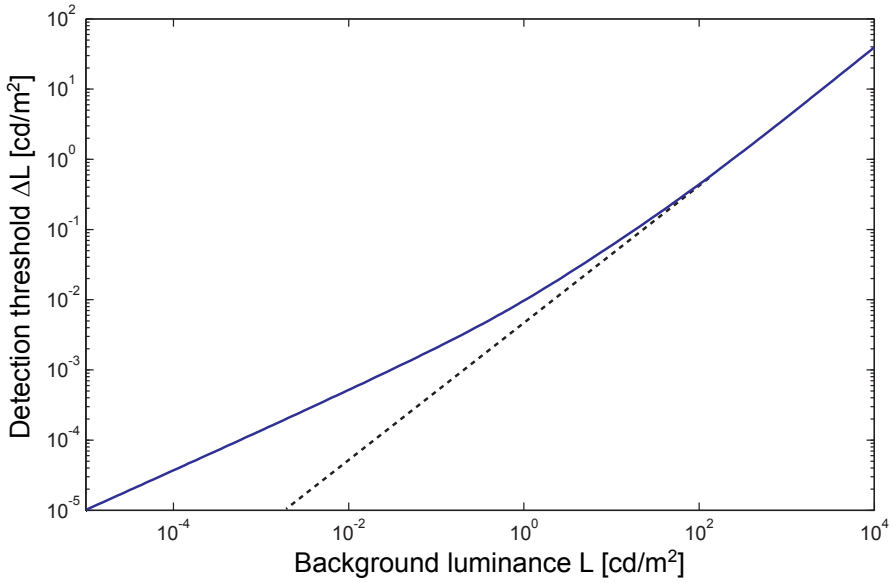


**Figure 2.7** – The threshold versus intensity (TVI) function

*Describes the minimum luminance difference that can be noticed on a uniform background luminance. Note that the function is approximately linear on a log-log plot. Dashed line indicates hypothetical detection thresholds for Weber's Law.* (Data from [Dal93])

composed of fast neural processes and relatively slow photochemical *bleaching* and *regeneration*. Bleaching occurs when we are exposed to bright light intensities and causes depletion of photosensitive pigments in the photoreceptors, thus reducing the sensitivity to the incoming light. After we decrease the light intensity, our sensitivity is gradually restored through the regeneration process. Compared to bleaching, the regeneration of photosensitive pigments is much more time consuming process, which is the main reason of asymmetry between the time course of adaptation to dark and bright stimuli. Conversely, neural processes are fast, symmetrical and result from the electrical response of photosensitive pigments to the light stimuli.

A fundamental measure of luminance adaptation is the TVI (*threshold versus intensity*) function [Hoo98], shown in Figure 2.7, that specifies the smallest perceivable light intensity (detection threshold $\Delta L$) as a function of background luminance $L$ upon which it was presented. Increasing the intensity of background changes our adaptation state and affects the sensitivity to luminance increments, such that a clear detection of light stimuli on a brighter background requires larger luminance difference between the stimuli and the background.

Current adaptation state or adaptation (background) luminance can be computed for the entire frame (globally) or for the specific area of the retinal image (locally). While it is not clear which mechanism dominates the computation of adaptation levels across the retina, in practice, for many applications the difference between the two is negligible and global

approach is chosen due to its computational efficiency [IFM05].

## 2.6 Contrast

An important consequence of the above considerations is that human visual system distinguishes objects not by the luminance difference alone, but by the difference in a relation to the background luminance. This visual property is known as *contrast*. Because our visual system is more sensitive to contrast than absolute luminance, we perceive the world with great detail regardless of large luminance ranges in various environments. The above definition of contrast is relatively loose and allows for multiple mathematical formulations specific to certain types of stimuli. For simple patterns, such as step function or foreground/background luminance, contrast can be defined as a *Weber's ratio*:

$$C = \frac{\Delta L}{L}, \tag{2.11}$$

which in context of digital images can be interpreted as a difference $\Delta L$ between the luminance of an image location with the luminance of certain neighborhood around that location, normalized again by the luminance of the same neighborhood.

For a sinusoidal luminance pattern, where we tend to perceive the grating as a whole, the contrast can be computed using Michelson's formula:

$$M = \frac{L_{max} - L_{min}}{L_{max} + L_{min}}, \tag{2.12}$$

with $L_{max}$ and $L_{min}$ being the sinusoid's peak points. Note that choosing among these "simple" contrast definitions is contextual, since they can trivially be converted to one another if required.

## 2.7 Contrast Sensitivity

In practical applications, where we often need to modify or directly compare contrast values, contrast metrics should be insensitive to the underlying background luminance, i.e. their response to stimuli with fixed contrast should remain constant over the entire range of acceptable background luminances. This characteristic, known as constant *contrast sensitivity*, can be partially validated for Equation 2.11 by looking at the TVI data. As shown in Figure 2.7, the function is linear over a range of approximately 3.5 log units of background luminances (starting from middle range), which suggests that the size of threshold

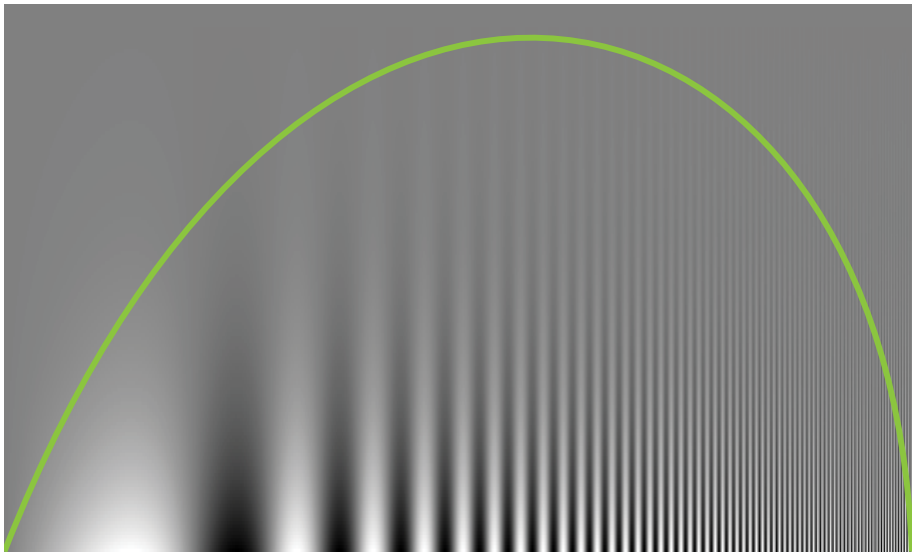

**Figure 2.8** – Campbell-Robson contrast sensitivity chart

*The luminance of pixels is modulated sinusoidally along the horizontal direction, with the frequency of modulation increasing exponentially from left to right. Additionally, the contrast magnitude is scaled down vertically to visualize various contrast levels. The green line indicates the smallest perceivable contrast value at a given frequency. The actual shape of the line depends on the viewing distance, type of display medium and the individual HVS characteristics of the human observer.*

increment increases proportionally to the background luminance. Since $\Delta L = kL$, then the contrast of a minimum detection threshold $\Delta L/L = k$ remains constant. Interestingly, this linear relationship between physical magnitudes and their perceived intensities is also valid for senses other than vision. In early 19th century, Ernst Heinrich Weber discovered that the just noticeable difference (JND) between two weights is roughly proportional to the mass of the weights.

Unfortunately, *Weber's Law* cannot be applied to HDR images, since it holds only for a limited range of adaptation luminances. It assumes constant contrast sensitivity, whereas the slope of TVI function decreases (Figure 2.7) when moving to mesopic and scotopic luminance ranges indicating that our eyes are less sensitive under dim illumination and a relatively larger luminance increment is required to create the same contrast impression.

The varying sensitivity to contrast stems from several limitations and trade-offs in design of our visual system (Section 2.2). There are approximately 125 million photoreceptors distributed non-uniformly (see Figure 2.4) across the retina. Before relaying all this data to the brain, the output of photoreceptors is spatially encoded by retinal ganglion cells. These cells are tuned to respond to contrast at spatial frequencies that match the size of their receptive fields. Figure 2.8 illustrates contrast sensitivity changes with respect to the frequency of the input sinusoidal stimuli. A computational model describing this relation is known as *Contrast Sensitivity Function*. CSF defines the sensitivity as an inverse of the threshold Michelson contrast (Equation 2.12) and depends not only on background

(adaptation) luminance, but also on the frequency of harmonic stimuli, its orientation, spatial extent, and eccentricity with respect to the fovea. Typically, CSF exhibits a band-pass characteristic with the peak sensitivity varying between 4 and 10 cycles per degree, depending on the current adaptation level. The high-frequency cut-off is caused by the optical limitations of the eye. Even if the quality and performance of eye's lens would be ignored, the ability to discriminate small features is bounded by the physical size of a single receptive cell. On the other hand, the low-frequency drop-off is a result of lateral inhibition within ganglion cells. The uniform luminance pattern falling on both inhibitory and excitatory regions of a receptive field leads to lateral inhibition and weak response of a ganglion cell.

There are a number of CSF models available [Dal93, Kel83, Bar99], each one having its own set of input parameters resulting from different experiment setup. The psychophysical experiments determining the CSFs have been performed on a near-threshold stimuli. For supra-threshold contrasts the CSF becomes flatter, indicating that our visual system is equally sensitive to all visible frequencies. This characteristic, known as *contrast constancy* [GS75], allows us to clearly see high-contrast objects regardless of their size or distance. In this thesis we use Daly's CSF [Dal93], since it proved to generate better predictions for HDR images and tolerates adaptation levels above $1000 cd/m^2$. The function is defined as follows:

$$CSF(\rho, L_a, \theta, i^2, d, c) = P \cdot min\left[S_1\left(\frac{\rho}{r_a \cdot r_c \cdot r_\theta}\right), S_1(\rho)\right], \tag{2.13}$$

where parameters

$$\begin{aligned}
r_a &= 0.856 \cdot d^{0.14}, \\
r_c &= \frac{1}{1+0.24c}, \\
r_\theta &= 0.11\cos(4\theta) + 0.11, \\
S_1(\rho) &= \left[(3.23(\rho^2 i^2)^{-0.3})^5 + 1\right]^{-\frac{1}{5}} \cdot A_l \varepsilon \rho e^{-(B_l \varepsilon \rho)} \sqrt{1 + 0.06 e^{B_l \varepsilon \rho}}, \\
A_l &= 0.801(1 + 0.7 L_a^{-1})^{-0.2}, \\
B_l &= 0.3(1 + 100 L_a^{-1})^{-0.15},
\end{aligned}$$

correspond to:

- $\rho$ – spatial frequency in cycles per visual degree $[cpd]$,
- $L_a$ – luminance adaptation level in $cd/m^2$,
- $\theta$ – orientation angle in degrees $[deg]$,

- $i^2$ – stimulus size in $deg^2$ ($i^2 = 1$),
- $d$ – distance in meters $[m]$,
- $c$ – eccentricity ($c = 0$),
- $\varepsilon$ – model's constant ($\varepsilon = 0.9$),
- $P$ – absolute peak sensitivity ($P = 250$).

Formulas for $A_l$ and $B_l$ contain corrections released after the initial publication ([Dal93]). Another viable option would be a recently introduced CSF by Mantiuk et al. [MKRH11], which has been carefully calibrated and is known to work well for HDR sources.

## 2.8 Perceptual Response to Light – Brightness

Brightness is defined as a perceptual response of our visual system to the retinal illuminance. The response to this attribute has been measured in the magnitude estimation experiments [SS63] in which the observers rated the brightness of the stimuli using a numerical scale 0-10. In conclusion, the study has reveled that the brightness relation to luminance can be described by a power function, with the exponent roughly equal to 1/3. Similar power function is used in LDR image representations to account for displays non-linear output characteristics. Therefore, despite a significant difference in exponent values ($\approx 0.45$ vs $\approx 0.33$), LDR pixels can be interpreted as a rough approximation of brightness.

Brightness can also be derived analytically from the Weber fraction contrast definition (Equation 2.11). Let $R(L)$ be a hypothetical response function of visual system to the retinal illuminance $L$. The difference in response should be equal to one for two luminance levels which are just noticeable:

$$R(L+\Delta L) - R(L) = 1 \iff \frac{\Delta L}{L} = k \tag{2.14}$$

Given the derivative $R'(L)$ we can find the response function by integration:

$$\begin{aligned} R'(L) &= \frac{R(L+\Delta L) - R(L)}{\Delta L} = \frac{1}{\Delta L} \\ R(L) &= \int_0^{\infty} \frac{1}{\Delta L}\,\mathrm{d}L, \quad \Delta L = kL \\ R(L) &= \int_0^{\infty} \frac{1}{kL}\,\mathrm{d}L = \frac{1}{k}\log(L) + k_1, \end{aligned} \tag{2.15}$$

where $k$ is a scaling constant resulting from Weber's Law and $k_1$ is a response offset, which

is usually set so that the response $R$ for the smallest detectable luminance $L_{min}$ is zero ($R(L_{min}) = 0$).

Both approximations (log-based and power-based) transform luminance into perceptually uniform units. Despite obvious difference in shape, they generate similar responses for luminance levels below $100 cd/m^2$, which is the most relevant range for majority of applications. However, in case of HDR images, where luminance often covers more than 3 orders of magnitude, the log-based transformation is preferable, since power function grows too rapidly for large luminances, distorting the relative importance of bright and dark image regions.

A more precise approximation has been introduced by [NR66], where they model the photoreceptor response to luminance (brightness) with a sigmoid curve. In contrast to the above models, the S-shape function is not limited in luminance range and incorporates the observer adaptation state:

$$R(L) = \frac{L^n}{L^n + \sigma^n} \cdot R_{max}, \tag{2.16}$$

where $R_{max}$ is the maximum response value, $\sigma$ is the half-saturation constant depending on the current adaptation state and $n$ controls the sensitivity and typically varies between 0.7 and 1. The curve is centered at the current adaptation level and shows a compressive behavior when moving away from the center. This suggests that the sensitivity peaks for the scene luminance equal to the current adaptation luminance. In such a scenario, our visual system is fully *adapted* to that scene luminance. Conversely, for luminance levels other than adaptation luminance, the photoreceptor response is compressed, and the visual system is *maladapted* to those luminances.

It should be noted that brightness models presented here are derived from responses of our visual system to a simple stimulus. For complex images, the brightness of a given point is more contextual and depends not only from corresponding luminance value, but also from the neighborhood contrast and overall image structure. This is demonstrated in Figure 2.9, where there is a visible difference between checkerboard patches, though their intensity is the same. A cast shadow dims the "brighter" patch, which is compensated by the visual system that recognizes the patch to be a part of larger surface. The apparent brightness of the patch is further improved by a local contrast between its neighbors, roughly corresponding to the contrast in the other patches. While this illusion demonstrates a faulty nature of our visual system as a physical light meter, it also shows that the visual system excels at high-level processing of visual information, allowing us to perceive objects as a "whole".

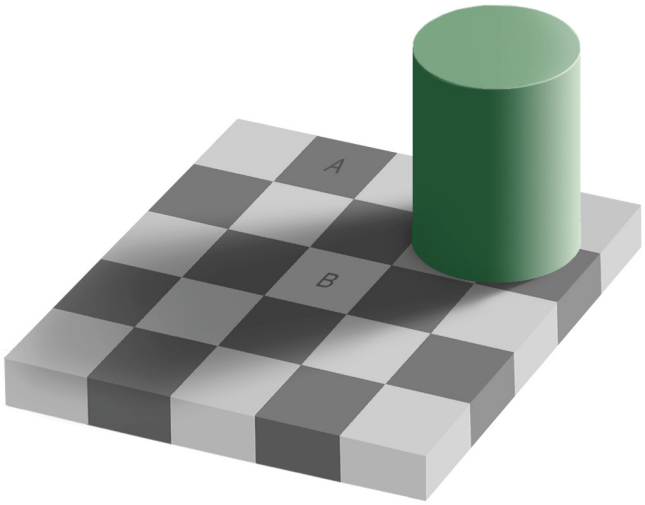


**Figure 2.9** – A checkerboard illusion image

*An example of contrast induced brightness sensation. When asked about the absolute brightness level of patches A and B, our typical answer will be that patch A is darker than B, but in fact, both patches have the same intensity.* (Image courtesy of Edward H. Adelson)

## 2.9 Visual Masking

It is generally known that we are most efficient at detecting spatial patterns when they are presented on a uniform background. As shown in Figure 2.10, sources of complex signals, such as natural images, can efficiently reduce the visibility of superimposed stimuli, especially when both signals are neighboring in space and frequency. This loss of sensitivity to a contrast patch ("target") in the presence of another stimuli ("masker") is referred to as *Visual* or *Contrast Masking*.

We can introduce a taxonomy that classifies masking models by the masking stimuli they consider [ZDL00]. First and most apparent form of masking – *self-masking* – occurs when both masker and target match in frequency and orientation, resulting in the target being masked by itself (see Figure 2.10). When signals diverge in frequency, masking effect decreases significantly, which improves the visibility of both stimuli. This effect is modeled by *inter-channel masking*, which accounts for masking between contrasts at the same spatial location, but in different frequency channels [LF80]. Finally, *neighborhood masking* refers to a drop of contrast sensitivity due to high activity of spatially adjacent neurons located in the same frequency channel [ZDL00]. High excitation of neighborhood neurons leads to lateral inhibition (see Section 2.7) and reduces neuron's sensitivity to absolute contrast values.

There are two dominant hypotheses explaining visual masking phenomena [WS97]. The first one assumes that the response to the contrast of a target is non-linear and compressive

**Figure 2.10** – An example of contrast masking

*The initial image (left) has been modified by adding a uniform sinusoidal distortion (right). Its visibility is reduced on the zebra, where stripes with similar size (frequency) mask the pattern. Note that the strength of the masking depends on a relative orientation between signals – the sine waves are still visible in areas where stripes are perpendicular to the distortions.* (Image courtesy of Tunç Ozan Aydın)

in nature [LF80]. Adding a masker with a similar frequency and orientation moves the response to the compressive range. Therefore, the difference between responses to integrated signals and the masker alone is reduced, leading to the elevation of visibility thresholds, as demonstrated in Figure 2.11. For a relatively large range of masker contrasts, when both signals match in frequency and orientation, the threshold value is proportional to the magnitude of contrast and thereby follows Weber's Law, $\Delta C = kC$. When a masker contrast is low, it facilitates the target rather than masking it. However, in practice this effect is often omitted, since its not as apparent as masking for complex images.

The second theory postulates the existence of inhibitory processes within contrast detection mechanisms [FPSG97]. The response of each neuron is divisively inhibited by a pool of responses of neurons located in a spatial and frequency neighborhood. More recently, models that combine both approaches have been proposed [Fol94, WS97].

Similar to brightness computation (see Section 2.8), we can linearize the perception of contrast and account for visual masking by applying a contrast *transducer* function. For example, the psychophysical model by Foley [Fol94] defines the transducer $R$ as follows:

$$\begin{aligned} R(C) &= E^p/(\sum_j I_j^q + Z), \\ E &= \max(0, \sum_i C_i s_{Ei}), \\ I_j &= \max(0, \sum_i C_{ij} s_{Iij}), \end{aligned} \tag{2.17}$$

where $E$ is the excitation factor that increases the response to contrasts, $I_j$ is the neural inhibition pool value for orientation $j$. Both are computed as a weighted sum with scaling

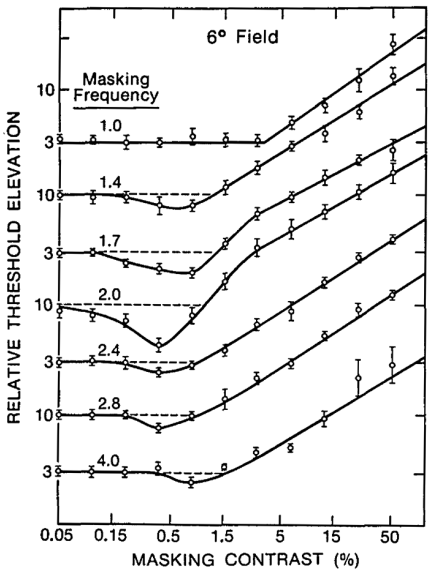


**Figure 2.11** – Threshold elevation as a function of masking contrast

*A sine-wave masking stimuli is displayed on top of a constant 2.0 cpd near-threshold sine-wave grating. For each masking frequency, a relative threshold elevation is plotted as a function of masking contrast (percentage of a target contrast). Note that when both signals are near-threshold the elevation is negative and leads to facilitation or "crispening" of the target signal. Dashed lines represent base detection threshold of a target.* (Image from [LF80])

coefficients $s_{Ei}$ and $s_{Ii}$ describing the sensitivity to contrast at spatial frequency $i$. A positive constant $Z$ controls the point where the saturation begins and prevents division by zero, while $p$ and $q$ exponents are used to fit the model to the experimental data. If the target signal is perceivable, but near the visibility threshold, the transducer should be scaled such that the differential between responses is equal to one JND (just noticeable difference):

$$\Delta R = R_{m+t} - R_m = 1, \tag{2.18}$$

where $R_{m+t}$ denotes a response of a combined masker and target and $R_m$ of the masker alone.

In this work we also employ transducer functions to account for perception of suprathreshold contrasts and to linearize their processing in the imaging pipeline. More comprehensive examples of these mechanisms are described in Section 4.3.4 (Equation 4.2) and Section 5.3.3 (Equation 5.3).

## 2.10 Multi-scale Decompositions

As discussed in Section 2.7, our visual system is tuned to certain contrast frequencies more than to the others. To predict contrast appearance at multiple frequencies, one needs to decompose the image into a set of band-pass channels, simulating the frequency selectivity of cortical cells. A precise solution to this problem is to use Gabor filter banks [Dau85]. Even though they faithfully model the on/off structures of receptive fields in visual cortex, high computational cost and lack of inverse transform makes the scheme impractical. Cortex Transform [Wat87, Dal93] represents a more viable alternative to Gabor filter, as it is invertible and offers a good trade-off between physiological plausibility and performance. On the other hand, for many image processing applications, a physiological precision is unnecessary, and therefore the strict channel separation requirement or the influence of contrast spatial orientation can be ignored. This fact is exploited in image enhancement [BA83, FFLS08, Fat09], compression [SCE01] or edge detection [MH80], where the band-pass decomposition is achieved through *multi-scale* methods that rely on fast linear filtering [BA83] or a wavelet analysis [URB97]. Remarkably, despite a rather approximate treatment of visual system characteristics, multi-scale methods dominate other approaches, since their flexibility and low computational overhead far outweigh the loss in precision. Such methods are also common in computer vision, where scale-space representation of the image enables vision systems to detect, extract and analyze features regardless of their scale [Lin98].

*Multi-scale decompositions* refer to algorithms that store and process images at multiple resolutions or signal frequencies (bands). Most of these frameworks feature a sort of a low-pass filter mechanism, whose main role is to gradually remove high frequency image features. As demonstrated in Figure 2.12, given a low-pass filter, we can efficiently compute a band-pass contrast representation in two steps. First, we generate a series of low-pass filtered images $L'$ with increasing blur level $\sigma$. Then, we compute a point-wise difference between neighboring blur levels to get the band-pass image. Provided that the input image contains perceptualized luminance values (brightness), such approach will produce a band-pass contrast measure. Using the transformation from Section 2.8, the process can written as:

$$\log(L'_{\sigma_1}) - \log(L'_{\sigma_2}) = \log\left(\frac{L'_{\sigma_1}}{L'_{\sigma_2}}\right), \tag{2.19}$$

which results in a *low-pass contrast* measure, defined as a logarithm of luminance ratio (logarithmic ratio).

Note that this approach will also work for regular 8-bit images, assuming gamma-corrected sRGB color space. As mentioned in Section 2.8, the power function accounting for display output non-linearity is roughly similar to log-based transform and can be used as an

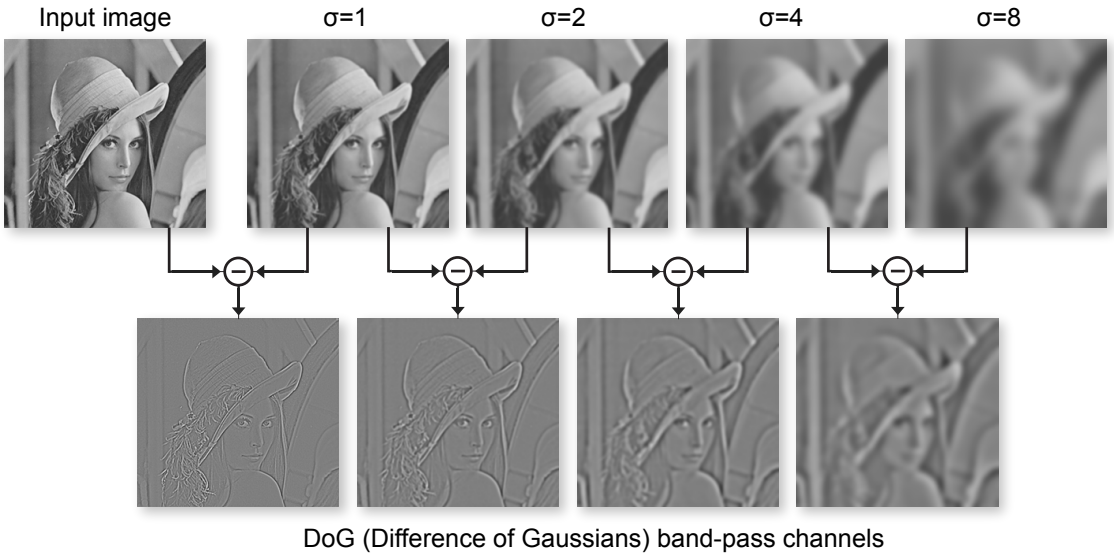


**Figure 2.12** – An example of multi-scale decomposition

*An input image has been decomposed into 4 band-pass contrast layers (bottom row) and an average luminance channel, often referred to as the DC component (top right-most image). Each contrast layer is computed as a difference between corresponding Gaussian blur levels (top row). The original image is reconstructed by adding all contrast channels and the DC component together. For visualization purposes contrast images are presented against a gray background.*

approximation of brightness perception for luminance levels below $100cd/m^2$. Therefore, the difference between pixels at scales $\sigma_1$ and $\sigma_2$ can be rewritten as:

$$L^{1/\gamma} \approx \log(L) \implies (L'_{\sigma_1})^{1/\gamma} - (L'_{\sigma_2})^{1/\gamma} \approx \log\left(\frac{L'_{\sigma_1}}{L'_{\sigma_2}}\right), \tag{2.20}$$

and similarly to the previous case, results in a logarithmic ratio contrast definition.

### 2.10.1 Gaussian Filtering

Since the filtering method is usually a key factor in the overall performance and quality of a decomposition framework, the majority of existing methods mimic the behavior of a *Gaussian* filter, which in its discrete form is known to produce good low-pass results [GW02]. For two-dimensional image data, the impulse response of a Gaussian filter is defined by:

$$K_\sigma(d) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{d^2}{2\sigma^2}\right), \tag{2.21}$$

where $d$ is the distance from the filter center (mean) and $\sigma$ denotes the standard deviation and controls the amount of blur. Conceptually, the $\sigma$ parameter can be interpreted as the minimum size of image features that will not be suppressed due to filtering.

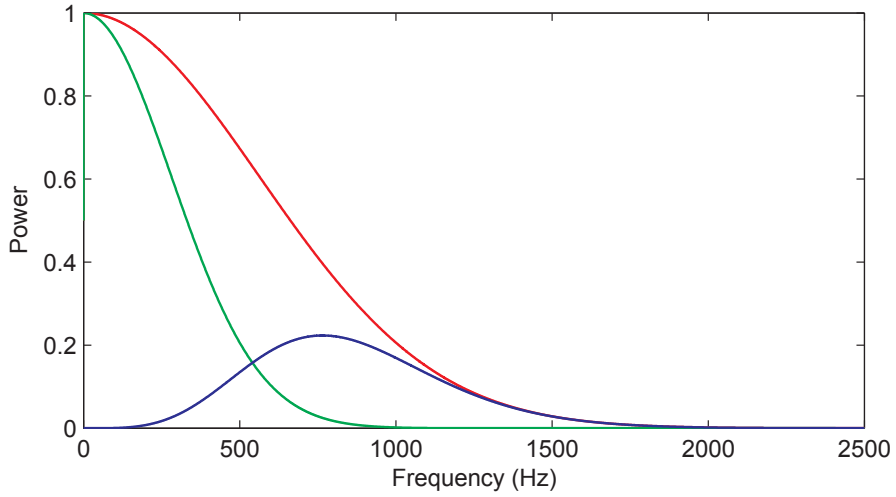


**Figure 2.13** – Power spectrum of a Gaussian filter

*A discrete impulse function has been blurred using a Gaussian kernel with σ values: 1 (red) and 2 (green). Increasing the σ narrows the filter bandwidth and improves its low-pass characteristics. The blue curve shows a DoG (Difference of Gaussians) output power spectrum – a band-pass output computed as a differential of two low-pass results. An identical result can be achieved by convolving the input function with a kernel defined as difference of corresponding Gaussian kernels.*

As shown in Figure 2.13, the frequency domain response of a Gaussian filter is a Gaussian function itself. Its frequency spectrum is smooth and band-limited, effectively reducing the range of scales over which intensity changes take place. Additionally, the non-negative and non-oscillatory nature of a Gaussian function prevents the introduction of new structures in the filtered image (e.g., ringing artifacts).

*Difference-of-Gaussians* (DoG) decomposition (Figure 2.12) uses the Gaussian filter to generate a band-pass contrast representation. The resulting difference of two Gaussian profiles approximates the Laplacian of a Gaussian, a simplified model of band-pass contrast perception [MH80]. Compared to more precise models, DoG maintains a weak band separation, causing each contrast band to receive a notable energy contribution from neighboring bands (Figure 2.13). We investigate this issue in a context of local contrast enhancement in Chapter 3.

A single low-pass channel $L'$ can be computed from the original image $L$ by means of a convolution with a specific Gaussian kernel $K_\sigma$:

$$L'(\cdot) = K_\sigma * L(\cdot), \tag{2.22}$$

where $*$ denotes the convolution operation. In a case of a symmetric Gaussian kernel, the convolution corresponds to a measure of cross-correlation, which tells us the amount of overlap of a Gaussian kernel $K_\sigma$ as it is shifted over the image $L$. In practice, the

implementation of a discrete convolution operator is realized with a sliding dot-product, and for a pixel $p$ is defined as follows:

$$L'(p) = \frac{1}{w(p)} \sum_{q \in \Omega} K_\sigma(||q||) L(p+q), \tag{2.23}$$
$$w(p) = \sum_{q \in \Omega} K_\sigma(||q||),$$

where $\Omega = \{(x,y) \in Z : -m \leq x \leq m \wedge -m \leq y \leq m\}$ and $K_\sigma$ denotes a discretized Gaussian kernel with a window spanning over $2m+1$ values in each direction. A normalization factor $w(p)$ ensures that $K_\sigma$ weights sum up to one. For an image with dimensions $w_{image} \times h_{image}$ the filtering is performed in $O(w_{image} h_{image} m^2)$ time, and becomes prohibitively expensive for large kernel sizes.

There are many techniques that reduce the computational complexity of Gaussian filtering [Sze10]. While the Gaussian function is always positive, the kernel weights decay rapidly when moving away from the window center. Therefore, to speed up the filtering without introducing any considerable errors, filter size is truncated to $6\sigma - 1$ values along each dimension. Additional performance boost is gained from the fact that Gaussian filter is *linearly separable*, that is, it can be applied to an image as two separate one-dimensional blurs, one in vertical and the other in horizontal direction.

Another fast Gaussian filtering method involves an iterative approach, where high image frequencies are successively suppressed with a box-filter. Implemented using a moving average algorithm [GW02], box-filter starts to approximate the Gaussian blur with as few as four iterations.

Finally, Gaussian filtering can be done directly in Fourier domain, where filtering of an arbitrary scale is performed in linear time. However, since the processing involves a relatively costly forward and inverse FFT transforms, it is recommended only in cases of large kernel sizes, for which the transform cost can be easily amortized.

### 2.10.2 Laplacian Pyramid

*Laplacian pyramid* [BA83] shares many similarities with DoG decomposition, but in contrast to the latter, the low-pass filtering stage is followed by a spatial subsampling of the image, resulting in a pyramidal shape of the structure. Each band-pass image is computed as a difference between current Gaussian blur level and its up-sampled successor from the level above. The scheme is based on the fact that the Gaussian filter with $\sigma$ equal to one removes half of the octave and thereby allows the image to be scaled down by a factor of two without introducing any artifacts. By repeating the procedure on a downscaled image we effectively double the $\sigma$ value every level, even though the actual filtering is performed

**Figure 2.14** – Detail enhancement with edge-preserving decompositions

*An input image (left) has been decomposed into multiple scales and had its medium and high frequency bands enhanced. Using a Gaussian kernel as a smoothing filter (middle) leads to visible "halo" artifacts around strong edges. This issue is addressed by edge-preserving filtering (right), that similar to a regular Gaussian filter, has a low-pass characteristics, but avoids smoothing across hard edges.*

with $\sigma = 1$ and $5 \times 5$ kernel. Laplacian pyramid improves on both performance and memory requirements, especially when compared to naïve scale-space implementations. However, it cannot be used in feature detection and other computer vision algorithms that depend on a band-pass phase signal. The phase (or shift) information in each band is lost due to the spatial subsampling of pyramid levels.

### 2.10.3 Edge-preserving Decompositions

In digital photography, artists tune contrast in selected areas to achieve certain aesthetical appearance of images. It has been shown that people prefer images with elevated contrast [YMMS06]. We especially favor sharp details, which is understandable given the low sensitivity of our visual system to high frequency contrasts (see Section 2.7). Also, the majority of natural images feature a power spectrum, in which most of the signal energy resides in low frequency bands [GB00]. Consequently, many pictures taken with digital cameras have poor visibility of details and require contrast enhancement to get satisfying appearance [YMMS06]. Since we do not perceive all frequencies equally well, editing contrast in multi-scale seems like a natural choice for the end user, and a particularly strong application of multi-scale decomposition frameworks.

In Figure 2.14, we demonstrate how a band-pass decomposition, such as Laplacian pyramid, can be used to enhance contrast in medium and high frequency bands. Although the results are generally plausible, the Laplacian pyramid generates visible distortions near hard edges. Such distortions, commonly referred to as "halo" artifacts, result from a smooth band-pass characteristics of a Gaussian filter, for which the response to a step function (edge) is spread over a large number of scales. *Edge-preserving* decompositions

address this issue by employing a low-pass filter that avoids smoothing across high-contrast edges.

The *bilateral filter* (Figure 2.14, right) is a classic example of a non-linear low-pass filter with an edge-preserving property [TM98]. In contrast to a regular Gaussian filter, it takes into account the difference between pixel intensities to preserve the edge while smoothing. The main idea is that a pixel should be averaged with neighbors who have a similar value, as most likely they belong to the same surface. Conversely, pixels occupying different intensity space are probably separated by an edge, hence their contribution to the smoothing should be reduced. We can extend Equation 2.23 to incorporate the notion of a bilateral filter, and define the resulting pixel $p$ value as:

$$L'(p) = \frac{1}{w(p)} \sum_{q \in \Omega} K_{\sigma_r}(L(p) - L(p+q)) K_{\sigma_s}(||q||) L(p+q), \tag{2.24}$$
$$w(p) = \sum_{q \in \Omega} K_{\sigma_r}(L(p) - L(p+q)) K_{\sigma_s}(||q||),$$

where smoothing kernel $K_{\sigma_s}$ is weighted by intensity distance matrix $K_{\sigma_r}$, both with window width of $2m+1$ elements for $\Omega = \{(x,y) \in Z : -m \leq x \leq m \wedge -m \leq y \leq m\}$. A normalization factor $w(p)$ ensures that the product of $K_{\sigma_r}$ and $K_{\sigma_s}$ sums up to one. The filtering process is controlled through two distinct parameters: $\sigma_s$ and $\sigma_r$. The strength of the spatial blur is adjusted with $\sigma_s$, whereas $\sigma_r$ reflects the tolerance to the difference in pixel intensity. Larger $\sigma_r$ widens and flattens $K_{\sigma_r}$, bringing the bilateral filter closer to a regular Gaussian filter.

Another type of filter is presented in the *weighted least squares* (WLS) decomposition framework [FFLS08]. There, the problem of an edge-preserving smoothing is viewed as a compromise between two contradictory goals. On the one hand, the resulting image $L'$ should be as close to the input $L$ as possible, but at the same time, we need $L'$ to be smooth everywhere except for the locations of strong edges. We can formally define the problem as a minimization of an objective function

$$E = \sum_{p \in \Omega} \left( (L'(p) - L(p))^2 + \lambda \left( a_{x,p}(L) \left( \frac{\partial L'}{\partial x}(p) \right)^2 + a_{y,p}(L) \left( \frac{\partial L'}{\partial y}(p) \right)^2 \right) \right), \tag{2.25}$$

where $\Omega = \{(x,y) \in Z : 0 \leq x < w_{image} \wedge 0 \leq y < h_{image}\}$. While the first term tries to keep the output $L'$ close to the input, the second term attempts to smooth $L'$ by minimizing its partial derivatives. The value of $\lambda$ balances both terms, and effectively provides a way to control the amount of blur. The filter preserves edges through weighting of the smoothness term by spatially variant $a_x$ and $a_y$, which depend on directional gradient magnitudes in

the luminance image $L$:

$$a_{x,p}(L) = \left( \left| \frac{\partial L}{\partial x}(p) \right|^{\alpha} + \varepsilon \right)^{-1} \qquad a_{y,p}(L) = \left( \left| \frac{\partial L}{\partial y}(p) \right|^{\alpha} + \varepsilon \right)^{-1}, \tag{2.26}$$

where $\alpha$ controls the sensitivity to gradients in $L$ and $\varepsilon$ prevents division by zero. After differentiation of $E$, the solution becomes coincident with a solution of an inhomogeneous Laplace equation, and can be efficiently obtained by solving a sparse linear system.

These, and other edge-preserving decompositions [BA83, MMS06, FFLS08, Fat09] are used extensively throughout this thesis. We have proposed a fast GPU-based implementation of selected state-of-the-art decomposition methods, and use it not only for image enhancement and manipulation in Chapters 3 and 5, but also for fast surface reconstruction in Chapter 4. For a more detailed discussion regarding these methods, we refer the reader to Chapter 3.

## 2.11 Visual Data Compression

In this section we provide a short introduction to the essential concepts behind modern image and video compression techniques. In particular, we focus on *lossy* video compression methods and their relation to human perception to which they owe their high compression efficiency. Note that we only highlight topics which are relevant to the understanding of Chapter 4. Recently released book by Richardson [Ric10], with an introduction to the state-of-the-art H.264 (MPEG-4 Part 10) video codec, provides a good complementary source to the ideas presented herein. Furthermore, we recommend Pennebaker [PM92] and Taubman [TM01] for a more detailed description of JPEG and JPEG2000 standards respectively.

*Video compression* or *video encoding*, is the process of reducing the amount of data used to represent the digital video signal. The inverse operation, *decompression* or *decoding*, deals with the reconstruction of a digital video from a compressed representation. Since digital videos require large amount of storage or bandwidth, video encoding and decoding, collectively referred to as video coding, is essential for any application with storage or transmission capacity constraints. A better video compression is the key to reducing the cost of delivery, which is especially important nowadays, when high definition content and high resolution displays are widespread.

The state-of-the-art video coding standards, such as H.264, owe their high compression efficiency to two important assumptions about the content. First is that the encoded data represents natural or real-world scenes, which are sampled both spatially and temporally.

Such scenes represent a variety of illumination conditions (see Figure 2.2) and may contain multiple objects, each having its own shape and texture. However, signals in each frame are usually continuous and follow certain patterns, like a specific shape of power spectrum or color distribution. Additionally, since we capture frames at relatively high speeds, the amount of information introduced in new frames is usually small, and mostly results from camera or object motion. This spatial and temporal redundancy is heavily exploited by video encoders that only encode the data that cannot be derived from the spatial or temporal neighborhood. The second assumption is that digital videos are intended for a human observer, whose visual system performance highly depends on characteristics of the input stimuli. Therefore, to limit the amount of information that is encoded, but would not be visible after the decoding, frames are compressed in a perceptually linear domain. Also, in scenarios where the video signal needs to be truncated due to bandwidth constraints, working in such domain allows to remove image features based on their visual importance. The resulting compression errors have a perceptually uniform distribution, which minimizes the visibility of potential artifacts.

However, if for some reason the video sequence violates any of the above assumptions, standard video encoders will perform poorly. This is shown in Chapter 4, where depth and motion data contains signals with spatial distribution different from the one found in natural images. Consequently, H.264 fails to precisely encode this kind of signal, which leads to compression artifacts in applications that rely on precise depth encoding. This becomes the primary motivation for our custom video encoding scheme.

Despite the tremendous amount of R&D put into development of efficient video compression algorithms, the majority of existing schemes still follows the classic MPEG pipeline, which dates back to 90s. Although this fact is mainly attributed to the requirement of backwards compatibility with existing hardware, the MPEG pipeline actually proved to be robust and very flexible [WSBL03]. Its latest adaptation, the H.264 Advanced Video Coding standard, delivers high quality video at half of the bandwidth used by its predecessor, MPEG-4 Part 2[1].

In Figure 2.15, we show a generalized model of H.264-class video encoder. The frame encoding order depends on the configuration of temporal prediction scheme (see Section 2.11.2), but for brevity, we can assume that frames are processed sequentially. The compression of frame $F_n$ proceeds as follows. Initially, $F_n$ is partitioned into a set of *macroblocks*, independently coded $16 \times 16$-pixel regions. For each macroblock $M$ we perform the following steps. First, we decorrelate its color channels by conversion to an opponent color space YCbCr, for which the compression can be performed on each color channel separately. Next, we form two predictions of $M$: $P_s$ and $P_t$. The spatial prediction $P_s$ is based on already encoded (or decoded at the decoder) neighbors of $M$: top, left, and top-

[1] Implemented by several codecs including DivX and Xvid

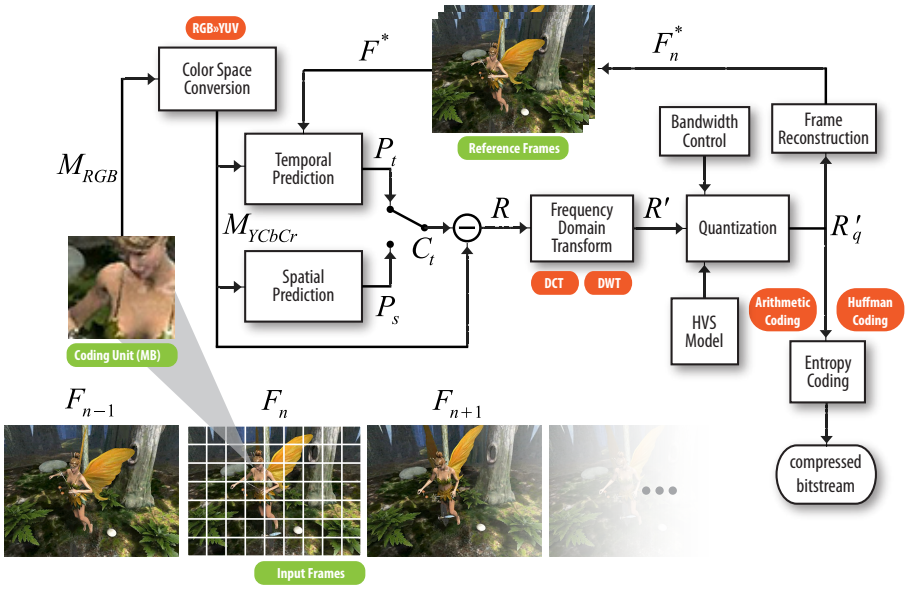


**Figure 2.15** – A generalized view of video encoding

*The majority of available video compression methods works alike. First, color channels of the input frame are converted to YUV space and divided into tiles (macroblocks) of equal size. Each macroblock is then decorrelated using a spatial or temporal predictor followed by a transformation to the frequency domain. The resulting coefficients are subject to quantization (lossy step) guided by human visual models and bandwidth constraints. The remaining data has little correlation and can be encoded efficiently using a loss-less statistical compression algorithms.*

right. On the other hand, the temporal prediction $P_t$, tries to find a good approximation of $M$ in a set of reference frames $F^*$. Out of these two predictions we select the one pointed by the cost function $C_t$, that estimates which predictor will result in a smaller and more "compressible" residual. The corresponding residual $R$ is formed by simply subtracting $M$ from "the best" prediction. Frequency domain transform, such as DCT (Discrete Cosine Transform) or DWT (Discrete Wavelet Transform), is used to remove the remaining spatial correlation within the residual macroblock. The resulting transform coefficients $R'$ are subject to non-uniform quantization, which depends on current bandwidth constraints. Quantized residual $R'$ is used in reconstruction of $F_n^*$ – a view of $F_n$ on the decoder side. Since we use past frames from $F^*$ as candidates for temporal prediction, both encoder and decoder have to use the same set of references to be able to reconstruct $F_n$ from $R'_q$. Finally, the quantized coefficients in $R'_q$ are transformed into a compact bitstream using an entropy coding method (see Section 2.11.5).

Surprisingly, except for the temporal prediction mechanisms, the compression pipeline for still pictures remains largely unchanged [PM92, TM01]. To compensate for the lack of temporal prediction, lossy image compression methods employ sophisticated visual models to account for luminance adaptation [Wat94] or contrast masking [ZDL00] (see Sections 2.5 and 2.9). These models significantly improve the coding efficiency while introducing a

notable increase in computational complexity. However, such increase in this case is tolerable, since compared to video coding, image coding applications are less constrained by computation time.

In the following sections we outline the details of particular MPEG compression steps.

### 2.11.1 Color Space Conversion

The first and perhaps the simplest method to reduce correlation in a macroblock $M$ is to convert sRGB color coordinates to an opponent color space, such as YCbCr. This can be realized as follows:

$$\begin{aligned} M_Y &= 0.299M_R + 0.587M_G + 0.114M_B, \\ M_{Cb} &= 0.564(M_B - M_Y), \\ M_{Cr} &= 0.713(M_R - M_Y), \end{aligned} \tag{2.27}$$

where $(M_R, M_G, M_B)$ is a color of a sRGB encoded pixel, $M_Y$ describes luminance and the $(M_{Cb}, M_{Cr})$ pair defines chrominance. YCbCr, similarly to sRGB color space, uses gamma-correction to optimize bit allocation in its integer representation. To avoid confusion with photometric luminance and chromacity, YCbCr components are referred to as *luma* and *chroma* respectively. The corresponding inverse transform is defined as:

$$\begin{aligned} M_R &= M_Y + 1.402M_{Cr}, \\ M_G &= M_Y - 0.344M_{Cb} - 0.714M_{Cr}, \\ M_B &= M_Y + 1.722M_{Cb}. \end{aligned} \tag{2.28}$$

As mentioned in Section 2.3, YCbCr represents colors in a perceptually uniform space, and therefore improves on the visual quality of the compressed output. Furthermore, since we are highly tuned to real-world scenes, such encoding seems suitable for videos following natural image statistics. Reduced dependency between channels allows to process them separately, and because we are less sensitive to color than to luminance, the chroma information is compressed more aggressively through stronger quantization and spatial subsampling. Table 2.1 summarizes common chroma subsampling modes and the effective pixel size in bits. Through simple linear transformation and spatial subsampling, we are able to cut the amount of visual information by half, even before the actual compression began.

In digital imaging and video systems, term YUV is often used to describe YCbCr, but in fact these are two separate color spaces. Historically, YUV was developed for analog television, mainly as a backwards compatible encoding for color and monochromatic tele-

| Pixel Format | Luma downsampling | | Chroma downsampling | | Effective BPP |
|---|---|---|---|---|---|
| | Horizontal | Vertical | Horizontal | Vertical | |
| YUV420 | 1 | 1 | 2 | 2 | 12 |
| YUV422 | 1 | 1 | 1 | 2 | 16 |
| YUV444 | 1 | 1 | 1 | 1 | 24 |

**Table 2.1** – Spatial subsampling schemes of YUV color space

*Here we show luma (Y) and chroma (UV) downsampling ratios for various YUV pixel formats. YUV420 stores chroma downsampled by a factor of two, but due to low sensitivity to the chrominance, the visual difference between YUV420 and YUV444 is negligible. Compared to a regular RGB encoding, where storing a single pixel requires 24 bits of memory, YUV420 uses half of the space, without introducing any significant drop in quality.*

vision systems. YCbCr, on the other hand, is used for digital encoding of color, as in MPEG or JPEG standards. Here we follow the convention and use YUV and YCbCr interchangeably, although in both cases we have YCbCr in mind.

### 2.11.2 Spatial and Temporal Prediction Schemes

The purpose of prediction schemes is to reduce the correlation of video data. The scenes captured with digital cameras usually contain continuous signals and many similarities between frames. In fact, most of the information about objects in current frame exist in previous or future frame in translated or scaled form. We can exploit this redundancy by building a prediction of macroblock $M$ based on pixels in current (spatial) or adjacent frames (temporal). The H.264 encoder implements spatial and temporal decorrelation separately.

The spatial prediction scheme, commonly referred to as *intra-prediction*, tries to anticipate the values of macroblock $M$ by inspecting only its adjacent neighbors: top, left and top-right. Since signals inside the frame vary smoothly, there is a high probability that neighboring macroblocks are similar to $M$. Therefore, instead of storing the macroblock directly, we can store the difference between $M$ and the prediction $P_s$.

The temporal prediction or *inter-prediction* scheme takes advantage of redundancy existing between frames. The key idea is that we can find a good estimate of $M$ by looking around its location in neighboring frames $F^*$. The resulting candidate might be translated with respect to $M$, likely due to the camera or object motion. The translation vector is referred to as "motion vector" and can be computed in a process called *motion estimation*. In general, motion estimation attempts to compute a function that transforms one frame to another, but MPEG coders use simpler approach based on block matching algorithms. The resulting prediction $P_t$ will be in most cases more precise than $P_s$. However, in contrast to the latter, where we just need to store the residual and a small number indicating the

selected spatial predictor, temporal prediction requires that together with the residual we encode a 2D motion vector and the reference frame number as well. Therefore, to reduce the encoding cost of motion vectors, H.264 applies intra- and inter-prediction to their values prior entropy coding.

Based on types of predictions that macroblocks are allowed to use, we can define 3 main classes of frames:

- I-frames – all macroblocks rely only on intra-prediction,
- P-frames – each macroblock can use intra-prediction or inter-prediction limited to past frames only,
- B-frames – each macroblock can use intra-prediction or inter-prediction based on both past and future frames (bidirectional prediction).

Note that, in order to use B-frames, the encoder has to compress and store frames out-of-order. For example, let us consider a frame sequence I(1)B(2)B(3)B(4)I(5), where numbers denote chronological order and letters the frame type. To decode frame number 3, we need to have frames 1 and 5, for frame number 2 we need frames 1 and 3, and so on. The resulting encoding order is defined as I(1)I(5)B(3)B(2)B(4), and leads to a 2 frame delay. Before we can show frame number 2, we need to read and decode frames 5 and 3. This is the main reason why, despite significantly improved compression, B-frames have a limited use in online applications. In web conferencing or online gaming, the playback delay is easily spotted, which adversely affects the user experience.

### 2.11.3 Frequency Domain Transform

The residual $R$ is defined as a difference between macroblock data $M$ and one of its predictions, $P_s$ or $P_t$. The quality of the prediction has a large impact on the overall compression performance. In many cases, particularly during high-motion sequences, the resulting residual complexity is still high and requires much space to encode. Therefore, video encoders further reduce the residual size by employing frequency domain transforms, which, as the name suggests, express the signal as a function of frequencies it contains. An example of such a transform is the Discrete Cosine Transform (DCT) that represents the data as a sum of cosine terms with varying frequencies. The DCT has two properties important for image and video compression. Firstly, it expresses the image signal in a domain where it can be efficiently represented with few coefficients (high energy compaction property). And secondly, the transformation is fully reversible and computationally tractable, i.e., has low-memory footprint and can be computed quickly, possibly using fixed-point integer arithmetic. The DCT is a block-based transform, and for the computational efficiency it is executed on relatively small blocks. For instance, the H.264 standard provides DCTs

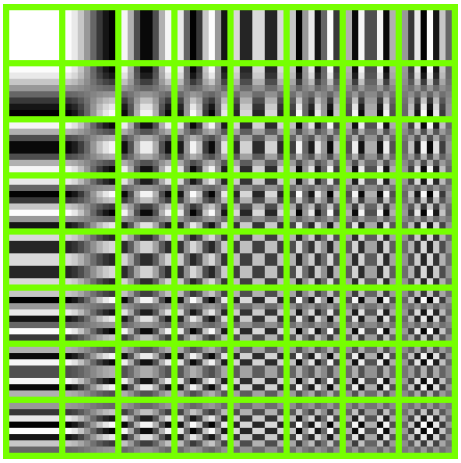

**Figure 2.16** – 2D DCT basis functions

*Here we show 64 basis functions of a 2D Discrete Cosine Transform (DCT) for* $8 \times 8$ *image block. Each basis function is a product of two cosines, with the frequency of oscillation increasing horizontally for the first cosine term, and vertically for the second one. Any kind of signal in the source block can be represented as a weighted sum of all basis functions, with weights corresponding to the DCT coefficients.*

for $8 \times 8$ and $4 \times 4$ blocks. Since each macroblock is $16 \times 16$-pixels big, the data inside has to be divided into smaller blocks before the transformation. The 2D DCT is defined as follows:

$$R'(i,j) = \sum_{y=0}^{N-1} \sum_{x=0}^{N-1} R(x,y) \cos\left(\frac{\pi}{N}(y+0.5)j\right) \cos\left(\frac{\pi}{N}(x+0.5)i\right), \tag{2.29}$$

where $0 \leq i < N \wedge 0 \leq j < N$ and $N$ denotes the block size. For a block of $N \times N$ pixels, the transformation produces $N \times N$ coefficients, each one corresponding to a basis function described by $i$ and $j$ coordinates, as shown in Figure 2.16. Mathematically, the transformation from Equation 2.29 is equal to the computation of a dot-product between block data and a particular basis function. Therefore, each DCT coefficient is a measure of similarity between the data and corresponding basis function. The value of $R'(0,0)$ represents the average signal in the block (Figure 2.16, top-left), and is often referred to as the *DC component*. The remaining coefficients, to emphasize their oscillatory nature, are collectively called the *AC component*.

The key factor behind the DCT performance is its strong energy compaction property. We need very few coefficients to represent the signal precisely, especially if the signal is of low-frequency nature. Such property is desirable in coding of a typical video content, for which the most of the image energy is located in low-frequency bands. In fact, if we would perform the principal component analysis (PCA) on a big set of natural images, the resulting eigenvectors would be similar to the DCT basis functions (see Figure 2.16). The H.264 uses the DCT to reduce the spatial redundancies of residual $R$, but the transform can be also applied to color channels directly, as shown in JPEG image coding standard [PM92].

Another frequency domain representation widely used in the compression is the Discrete Wavelet Transform (DWT) [JO05, TM01]. In contrast to the DCT, the wavelet transform is performed on the whole image. This allows to avoid artifacts that appear at block boundaries for very low bit-rates (high compression). Although it is successfully used in JPEG2000 image coder, the application of DWT to video coding seems limited to scenarios where very high-quality or near lossless compression is required [LOL97, BGG09]. One notable exception is Dirac video compression [dir08], that in contrast to MPEG-like encoders, uses DWT for residual coding.

### 2.11.4 Quantization

Quantization is a process of rounding a fractional number to the nearest integer. In video coding, it is used as a primary tool for controlling the bandwidth and visual quality of the compressed video. Although residual coding followed by the DCT removes most of the redundant signals within the frame, we can still improve the compression rates by lowering the variation of symbols in $R'$. This step, however, is irreversible and results in visual quality degradation. The perceived quality of output videos can be measured with many image and video quality assessment methods [AvMS10].

The quantization of transform coefficients in $R'$ is defined as follows:

$$R'_q(i,j) = \left\lfloor \frac{R'(i,j)}{Q_p(i,j)} + 0.5 \right\rfloor, \tag{2.30}$$

where $Q_p$ denotes the quantization matrix for compression level $p$, and $(i,j)$ describes the position in macroblock $R'$. Matrices with larger $p$ introduce more artifacts, but also make $R'_q$ compress better. The H.264 standard allows to define $p$ on a per macroblock basis.

The simplest approach is to quantize all coefficients uniformly. However, as mentioned in Section 2.7, the human visual system exhibits a varying sensitivity to contrast for different spatial frequencies. We can distinguish many levels of contrasts when they are spread over a relatively large area, but much fewer when they are concentrated in one location. Since the AC part of DCT coefficients could be thought of as an approximation of band-pass contrast, image and video encoders apply a non-uniform quantization to account for the shape of CSF. For example, the JPEG standard [PM92] defines the default quantization

matrices for luma and chroma as:

$$Q_Y = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}, \quad Q_{CbCr} = \begin{bmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 47 & 66 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{bmatrix}.$$

For matrix $Q_Y$, the coefficients representing low-frequency contrasts, to which we are most sensitive, are quantized less severe. When moving to higher frequencies, the quantization increases, indicating the drop of sensitivity. This effect is more pronounced in a case of $Q_{CbCr}$, where flat quantization of almost all frequencies reflects the overall low sensitivity to chroma.

Since the shape of CSF changes with background luminance, methods that extend JPEG by accounting for adaptation luminance and self-masking have been proposed [Wat94]. However, due to low spatial scalability of JPEG standard, which allows for only one quantization matrix per color channel, the performance of these schemes is questionable. Much better results of contrast masking and CSF modeling were reported with JPEG2000 coder, whose DWT transform is more suitable for scaling in both frequency and spatial domains [ZDL00].

### 2.11.5 Entropy Coding

Once we have removed all the redundant information from the frame, we would like to store the resulting residual $R'_q$ in a compact form. As described in previous sections, the residual data may contain quantized transform coefficients, motion vectors, macroblock headers, frame headers, etc. Video codecs employ an *entropy coder* to convert this data into a compressed bitstream suitable for transmission or storage.

An *entropy coder* is a loss-less data compression scheme designed to exploit statistical properties of the input stream [SM10]. It works by representing symbols in the stream as a variable length binary codes. Each code has a length roughly proportional to a negative logarithm of its probability, which indicates that more frequent symbols will require less bits to encode. Since previous encoding stages aimed at removing the correlation between and within frames, the residual $R'_q$ should contain mostly zeros or values close to zero. Video encoders benefit from such distribution and use entropy coding to reduce the bit length of common symbols.

Many image and video compression algorithms employ *Huffman coding* to compress the residual [PM92, Ric10]. Huffman coding is a form of prefix coding, a coding type where no

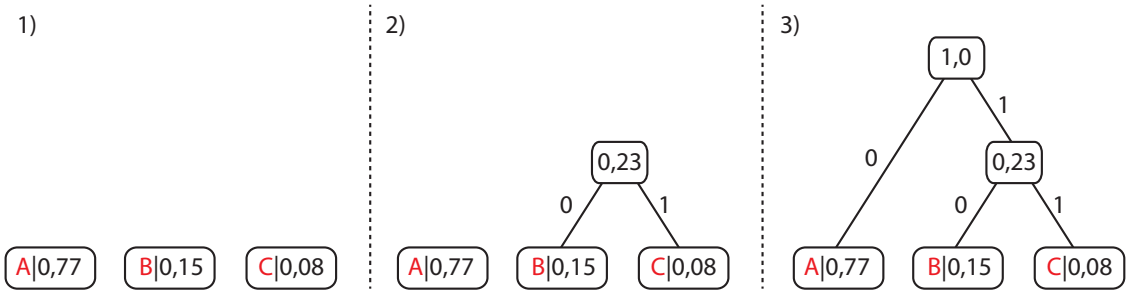


**Figure 2.17** – Construction of Huffman codes

*Huffman codes can be efficiently generated with a binary tree (Huffman tree). 1) We start by putting symbols and their probabilities into leafs. 2) The tree is created branch by branch, and in each step we add a node that links two elements (leafs or nodes) with the smallest probability. We set the probability of a new node as a sum of probabilities of its children. The process is repeated until the root node is created. 3) The final code of a symbol is defined by a path from the root to the corresponding leaf.*

symbol is a prefix (start) of any other symbol in a set. This property allows the decoder to identify each symbol without knowing its length.

For the sake of clarity, let us consider a simple example. Message $M =$ "*BAAAAABAAAAAC*", has 13 characters and contains 3 unique symbols $S = \{A, B, C\}$, where symbol $C$ is always put at the end of the stream to indicate end-of-data. The frequencies and corresponding probabilities of symbols are shown in Table 2.2, while in Figure 2.17 we visualize the construction of Huffman codes for $S$. Actually, the code generation accounts for the majority of the encoding step. To create a compressed message $M'$, we simply write out every symbol from $M$ using its corresponding Huffman code. The size of $M'$ is defined as follows:

$$M'_{size} = \sum_{s \in S} c_s ||h_s||, \tag{2.31}$$

where $||h_s||$ denotes the bit length of $h_s$. In a case of our example $M$ , the compressed message $M' =$ "1000000100000011" is 16 bits long. Now, to reconstruct the original message, except $M'$, we also need some additional information which will suggest how to reconstruct the tree from Figure 2.17. The decompression of each symbol is done by traversing down the Huffman tree until a leaf node is reached. The decision of whether to select left or right node is based on values of bits read sequentially from $M'$. When we reach a leaf node, we output its symbol and repeat the procedure until end-of-data symbol is decoded.

Another algorithm widely used in entropy coding is an *arithmetic coder* [WNC87]. In contrast to previous method, where each symbol is a unique prefix code and the message is coded symbol-by-symbol, arithmetic coding represents the entire message as a single number $M'' \in [0, 1)$. As shown in Figure 2.18, the encoding process is equivalent to a successive narrowing of the range in which the number can exist. After all symbols have been processed, any number within the computed range, together with the probability

| Symbol | Frequency $c_s$ | Probability $p_s$ | $-\log_2(p_s)$ | Huffman code $h_s$ |
|---|---|---|---|---|
| A | 10 | 0.769 | 0.379 | 0 |
| B | 2 | 0.154 | 2.7 | 10 |
| C | 1 | 0.077 | 3.7 | 11 |

**Table 2.2** – Symbol probabilities for message $M$ and their corresponding Huffman codes

model, will allow us to unambiguously identify the input sequence. The decoding step is very similar to encoding, and begins with dividing the initial range $[0,1)$ proportionally to symbol probabilities. Next, we check to which of the subranges $M''$ falls into. After we have determined the subrange, we output the corresponding symbol, and compute new range as illustrated in Figure 2.18. The process is repeated until end-of-data symbol (C) is written out.

For any given set of symbols and their probabilities, arithmetic coding approaches near-optimal output, and therefore follows Shannon's source coding theorem, which specifies $-\log_2(p_s)$ to be the optimal binary code length for a symbol with probability $p_s$. Consequently, the total size of compressed message $M''$ can be defined as:

$$M''_{size} = -\sum_{s \in S} c_s \log_2(p_s). \tag{2.32}$$

Given the probabilities from Table 2.2, the encoding of $M$ produces an interval $[0.80269, 0.80282]$. The minimum bitstream describing the message is equal to "110011011", and corresponds to $M'' = 0.80273$. Surprisingly, the stream is only 9 bits long, a value smaller than the theoretical limit of approximately 13 bits. This discrepancy results from the fact that Equation 2.32 defines an upper bound for the message size, whereas we encode the shortest number falling within the final interval.

The output message $M''$ is much smaller than the result of Huffman coding, where it takes 16 bits to store the same message. However, the superior compression performance of arithmetic coding comes at a price of increased computational complexity. To improve both encoding and decoding speeds, fast implementations rely on a fixed-point range computation, which includes only addition and bit-shift operations.

The main reason why arithmetic coding performs better is that it is a block entropy coder (all symbols are represented with one binary number), while Huffman coding is a scalar entropy coder (symbol-by-symbol). Furthermore, it offers the ability to adapt the probability model as the probabilities of symbols change w.r.t. time. In other words, instead of using a fixed probability model to compute the range of a symbol, we can use any model reproducible by both encoder and decoder. The simplest models involve accumulation of symbols up to present, i.e., we start with equal probabilities, and then

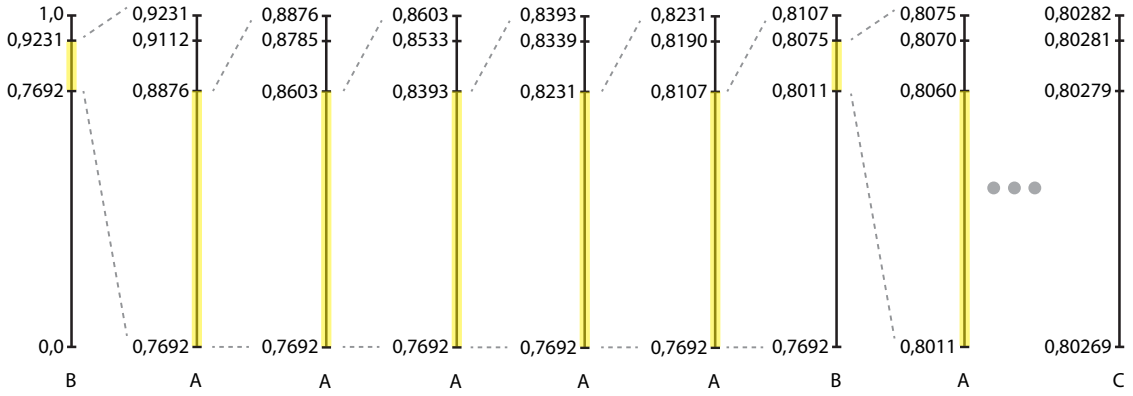


**Figure 2.18** – An example of arithmetic coding

*To find $M''$ that represents all symbols in $M$, we apply a successive "tightening" of the initial $[0, 1)$ range. First, for each symbol in $M$, we divide its range into subranges proportional to symbol probabilities (from Table 2.2). Then a subrange corresponding to the current symbol is selected as a range for the next one. The procedure is repeated until the end-of-data symbol (C) is encoded.*

update them after each encoded or decoded symbol. However, this method fails to capture local statistics, and only considers the long-term average. Much more efficient approach is to utilize a *local context* in computation of probabilities. For example, if a previously encoded symbol is used as a context, we can define $n$ probability models, where each model will adapt its probabilities to symbols existing next to a particular previous element type. This effectively models a *conditional probability*, and allows video codecs to reduce the remaining correlation in the residual frame data.

Although adaptive approaches for Huffman coding exist [Vit87], compared to arithmetic coders, they still offer lower compression performance. Additionally, since the cost of constructing codes dynamically is high, they loose the only advantage over arithmetic coders – encoding and decoding performance.

In Chapter 4, we present context adaptive arithmetic coding as a primary tool to efficiently encode images containing regular edge structures. There, to better exploit similarities between frames, we have defined the context as a mixture of spatial and temporal neighborhood.

## 2.12 Summary

In this chapter we presented current state of knowledge concerning modeling of human visual system and its basic characteristics, such as luminance adaptation, contrast sensitivity and visual masking. We showed how this knowledge is applied to state-of-the-art lossy video compression, image enhancement and visualization methods. Despite all the benefits

following the application of perception-based contrast models, their use in computer graphics algorithms is limited so far. This is surprising, since images and animations produced by computer graphics methods are intended for human observers. Currently, contrast is often represented, probably out of convenience, as a difference of gamma-corrected pixels – a representation that dates back to beginnings of computer graphics. Although it models few important aspects of our vision, including brightness perception (luminance adaptation), it is far from modeling a precise response of our visual system. Moreover, as shown in Figure 2.9, working directly on pixel intensities can be misleading, which emphasizes the importance of perceptual contrast processing. In the following chapters we will attempt to address this issue, and show the use of visual models presented here to improve the quality and computational efficiency in selected computer graphics applications.

# Contrast Prescription for Multi-scale Image Editing


**Figure 3.1** – Prescription idea in multi-scale contrast manipulation

*Enhancement of medium/high frequency contrast bands produces saturation of some image features existing in lower bands. Prescribing the contrast of unmodified bands prevents the saturation and at the same time allows to effectively increase the contrast according to user's request.*

Recently proposed edge-preserving multi-scale image decompositions enable artifact-free and visually appealing image editing. As the human eye is sensitive to contrast, per-band contrast manipulation is a natural way of image editing. However, contrast modification in one band usually affects contrasts in other bands, which is not intuitive for the user. In practice, the desired image appearance is achieved through an iterative editing process, which often requires fine tuning of contrast in one band several times. In this chapter we show an analysis of properties of multi-scale contrast editing frameworks and we introduce the concept of contrast prescription, which enables the user to lock the contrast in selected areas and bands and make it immune to contrast manipulations in other bands.

## 3.1 Introduction

Contrast editing is a common post-processing step in digital photography, usually aimed towards improving the photograph's aesthetical appeal. Research on contrast editing has been focused on two main issues: developing versatile, yet computationally efficient frameworks that produce artifact-free results, and developing user interfaces that provide intuitive interaction with these underlying frameworks. Common to all state-of-the-art contrast editing methods is the involvement of a multi-scale image decomposition, through

which the image contrast can be edited in an arbitrary number of scales. This tendency is not surprising; since the human visual system (HVS) comprises mechanisms to perceive contrast in multiple spatial frequencies, editing fine and coarse image details separately feels only natural for the end user.

In a multi-scale framework, a perfect separation between individual scales such that no two scales have any overlap, can theoretically be achieved by using frequency domain filters with sharp cutoffs. However this simple approach is never applied in practice since it results in heavy ringing artifacts, which can only be avoided by a smooth transition between filters of different scales. As a direct result, an important, but often ignored property of multi-scale frameworks is the interaction between contrast at individual scales. Thus, enhancing the contrast of, for example, medium frequency details, indirectly affects the appearance of fine and coarse details due to the overlap between the filters of the neighboring scales in frequency domain. The central idea of this work is "contrast prescription", where the user selects a certain image region for which the contrast at each unmodified scale is locked ("prescribed"), and thus the image details with the desired spatial frequency can be edited independently.

The practical implication of contrast prescription is a more intuitive contrast editing experience. While a set of user controls (often sliders) that control the contrast amplitude at different scales gives the impression of being orthogonal to each other, in reality the changes at one scale propagates to others; an effect which we call "leaking" of contrast. These leaks can effectively be prevented by prescribing the contrast in the image region being edited, which frees the user from iteratively adjusting interface controls to confine the contrast change to the desired scale.

We show that contrast prescription can be implemented in multiple state-of-the-art contrast editing frameworks. Our GPU implementation combined with an intuitive user interface comprising brush and slider controls provides real-time feedback. In this work we focus on editing both low dynamic range (LDR) and high dynamic range (HDR) images using ordinary (LDR) display devices. In the rest of the chapter we discuss related work on contrast editing and multi-scale image decompositions (Section 3.2), give details on the related consequences of multi-scale editing (Section 3.3) and how we address them (Section 3.4). Next, we discuss details on the extension of multiple frameworks to handle contrast prescription (Section 3.5), and finally present our results (Section 3.6).

## 3.2 Related Work

Multi-scale image decompositions such as the Laplacian pyramid [BA83, GW02] have been successfully applied to many image processing tasks, including image editing. The practical advantage of considering multiple scales for image editing is the ability to modify the appearance of coarse and fine details separately [LSA05]. On the downside, enhancing fine details disproportionally to coarser details leads in the extreme case to the well-known halo artifacts, resulting in unnatural images often considered not to be aesthetically pleasing.

*Edge-preserving image decompositions*, on the other hand, minimize halo artifacts by avoiding smoothing across strong edges. The edge preserving behavior is accomplished through non-linear filters such as weighted least squares [LBB88], anisotropic diffusion [PM90, BSMH98], or the bilateral filter [TM98]. Motivated by the anisotropic diffusion, Tumblin et al. [TT99] proposed a hierarchical approach called LCIS for HDR tone mapping purposes. The bilateral filter has been widely used in HDR tone mapping as well [DD02], but also in image fusion [ED04, FAR07], example-based transfer of photographic look [BPD06], among others. However, the extra complexity of the filters confine applications of Bilateral filtering to work offline. The performance issue has been addressed by introducing the "Bilateral grid" as a data structure built on Bilateral filter, which enables real-time, multi-scale edge-aware image manipulations [CPD07]. The edge preserving behavior has been further improved by a weighted least squares (WLS) based framework [FFLS08], and later another framework based on edge avoiding wavelets [Fat09] has been shown to achieve similar quality results much faster, due to the involvement of the linear time lifting scheme. The principle idea of preserving edges during decomposition has also been used in contrast processing of HDR images [MMS06]. This method relies on performing image editing by first scaling perceptually linearised image gradients, and then reconstructing the image from the new gradient values. Recently, Subr et al. [SSD09] proposed another edge-preserving image decomposition based on local extrema.

One consequence of multi-scale image editing is an increased complexity of the editing process from the user's perspective. In fact, interfaces for intuitive image manipulation have been an active topic of research [LSTS04, LLW04, LAA08]. In our work we rely on an intuitive brush based interface and set of scale selection sliders. With such a setup, we were able to generate the results in several short sessions. This is also attributed to our fast GPU-based implementation that provides a real-time feedback of the underlying contrast processing framework.

## 3.3 Consequences of Band Modification in Multi-scale Decompositions

Contrast can vaguely be described as the difference between the intensity of an image location with the intensity of some neighborhood around that location, normalized again by the intensity of the same neighborhood. A mathematical formulation of this quantity is possible for simple luminance patterns (such as a foreground-background stimulus with luminance profile defined by a step function) where contrast can be defined as Weber's ratio:

$$W = (L_{max} - L_{min}) / L_{min}, \tag{3.1}$$

or the logarithmic ratio

$$G = \log (L_{max} / L_{min}), \tag{3.2}$$

where $L_{max}$ and $L_{min}$ are the intensities of the foreground and the background, respectively. For a sinusoidal luminance pattern, contrast can be computed using Michelson's formula $M = (L_{max} - L_{min})/(L_{max} + L_{min})$ with $L_{max}$ and $L_{min}$ being the sinusoid's peak points. Note that choosing among these "simple" contrast definitions is contextual, since they can trivially be converted to one another if required.

Natural images, however, are much more complex than mere step or sinusoidal intensity patterns, in that they contain various details at multiple scales. Consequently, the computation of the aforementioned "simple" contrast measures is not clear since $L_{max}$ and $L_{min}$ are not well-defined. [Pel90] defines contrast in complex images as the ratio of the bandpass image to the low-pass image at multiple scales

$$C_i = \frac{K_{\sigma(i)} * I - K_{\sigma(i+1)} * I}{K_{\sigma(i+1)} * I}, \tag{3.3}$$

where $*$ denotes the convolution operation between linear luminance and a low-pass Gaussian kernel $K_{\sigma(i)}$ at scale $i$, and $\sigma(i) = 2^i/\sqrt{2}$ denotes standard deviation, which accounts for frequency band cutoff.

In this work, we use a multi-scale contrast representation, where each contrast sub-band is calculated as a ratio between successive ($i$ and $i+1$) Gaussian-like[1] smoothings of the image:

$$G_i = \frac{\text{smoothingOperator}(I, i)}{\text{smoothingOperator}(I, i+1)}. \tag{3.4}$$

To simplify the computations, the decomposition is performed on the logarithm of luminance $I$ (roughly approximating the non-linear perception of luminance), for which the ratio can be replaced with a simple subtraction. Such a difference then gives the loga-

[1] In practice, any type of low-pass (and also edge-preserving) filter can be used as a replacement of Gaussian filter.

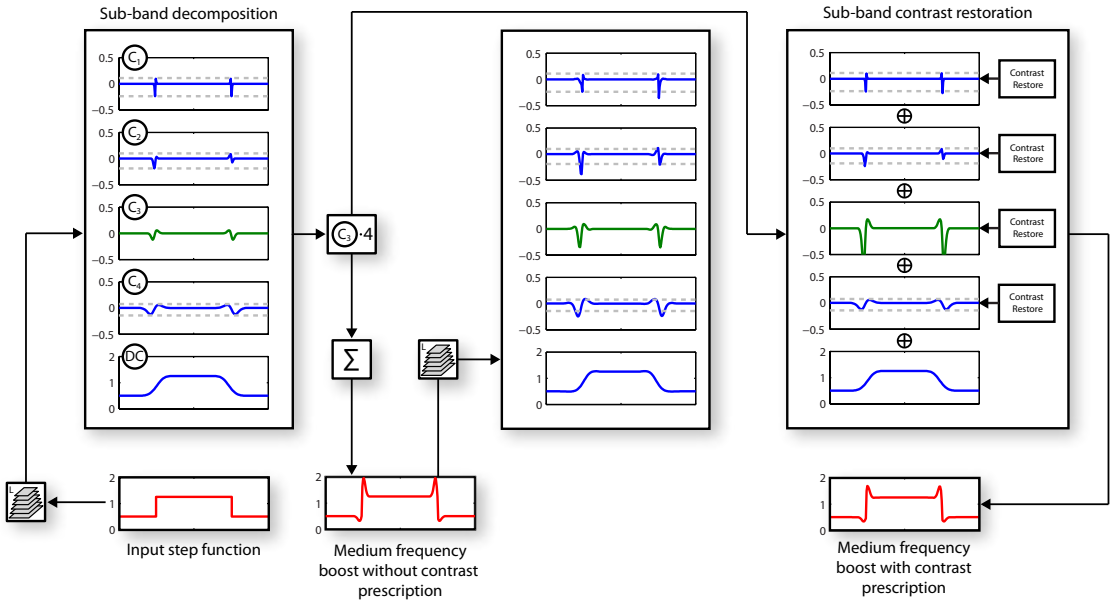


**Figure 3.2** – Step function contrast enhancement

*Input function is decomposed into 4 frequency bands and a DC component. Each band is visualized using Peli's definition for the physical contrast. We simulate detail enhancement by multiplying the $C_3$ band by 4. Decomposing the modified signal again shows that the contrast change of modified band is not proportional to the applied multiplier (middle column). Furthermore, due to Gaussian filter* energy leaking *property, all neighboring bands have been modified. Contrast prescription during the computation of modified signal (right column),* locally *limits the energy leaking and allows for better preservation of the contrast values in unaffected bands.*

rithmic ratio between an image location at some scale and the mean of its neighborhood, as shown in Equation 3.2. This representation has also the advantage of being computationally more efficient then Peli's contrast, therefore most of the multi-scale image editing frameworks [LFUS06, Fat09] follow this simple band decomposition scheme.

The *selection of smoothing operator* is usually the key factor in the overall performance and quality of a decomposition framework. Marr and Hildreth [MH80] define two main requirements that need to be met for the good smoothing filter. The first is that every multi-scale decomposition requires the filter spectrum to be smooth and band-limited in the frequency domain. This allows to reduce the range of scales over which intensity changes take place. We can design a band-pass filter that would be perfectly localized in the frequency domain (sharp cutoff or brick-wall type of filter). However, processing an image with such a filter will induce well known ringing artifacts. Non-oscillating low frequency parts of the image will yield in global oscillations in the output band representation. To prevent such artifacts, one needs to put the second requirement, a spatial localization constraint on the filter characteristics. This requirement, much more important from image editing point-of-view, can be interpreted in a way that every pixel in the filtered

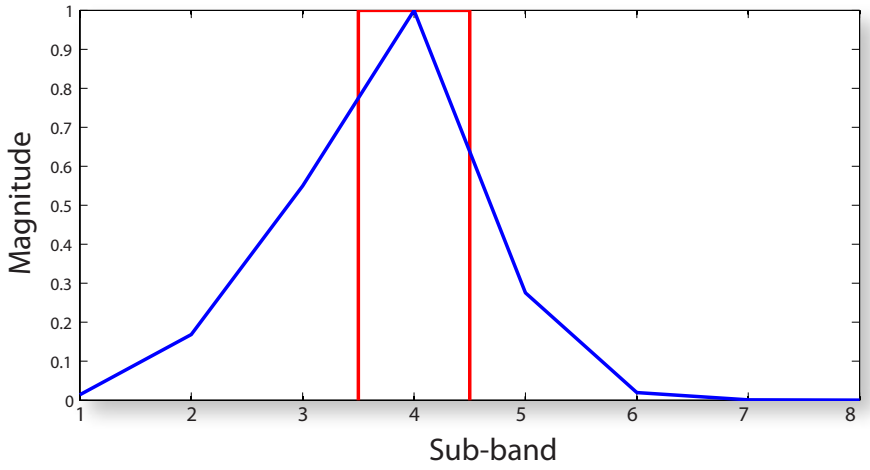


**Figure 3.3** – Comparison of *energy leakage* between low-pass filters

*A Guassian-based low-pass filter is compared against an ideal band-pass filter based decomposition. Input impulse signal is decomposed into 8 DoG (Difference-of-Gaussians) bands. While modifying the band we measure the signal magnitude change at each scale, which leads to piecewise linear approximation (blue plot). Due to non-ideal band-pass characteristics of the Gaussian filter, boosting the middle band results in uncontrolled amplification of features in neighboring scales (blue plot). The energy distribution of DoG decomposition resembles Gaussian function itself, but is not symmetric in logarithmic frequency scale. On the contrary, ideal band-pass filter satisfies the localization requirement in frequency domain (red). However, when applied to discrete image edges, creates undesired ringing effect in the spatial domain.*

image should be computed from a weighted average of nearby pixels. The constraints above are contradicting in a sense that we are able to increase the spatial locality (reduce ringing) at the cost of frequency domain performance (reduced band-pass behavior).

A good example of such a trade-off in filter design is the Gaussian low-pass filter. It is non-negative and non-oscillatory, hence causes no ringing. The response in the frequency domain is a Gaussian function itself with the mean focused around the middle frequency of the band. This feature has an important consequence when applied in multi-scale image processing. Such filter is inevitably causing an *energy leakage* between bands: the energy (modification) in one band leaks out to neighboring bands in the successive multi-scale image manipulations. We show an illustrative band manipulation example exploiting Gaussian filter in Figure 3.2 (left column), where single sub-band modification generates undesired energy leaking in neighboring sub-bands (Figure 3.2, middle column). The energy leakage is prevented when ideal band-pass filter is used (identified by box function in frequency domain), however it results in ringing artifacts as described above. In Figure 3.3 we compare the energy leakage of Gaussian-based image decomposition with this "ideal" band-pass decomposition.

The uniform smoothing behavior of the Gaussian filter, which leads to halo artifacts if sub-bands are independently modified, is addressed by so called edge-preserving smoothing

operators. They preserve sharp edges by excluding pixels across image discontinuities from consideration, which avoids dividing the energy of the same edge across multiple sub-bands. However, due to imperfect localization in the spatial and frequency domain of the low-pass filter, these halo-free decompositions are still affected by the inter-band energy leakage during band manipulation.

We are not aware of any practical image decomposition scheme which is free of inefficiencies of Gaussian-like low-pass filtering mentioned here. To address this problem we introduce *contrast prescription* which restores unmodified bands physical contrast (i.e. Peli's contrast) during image reconstruction process (see Figure 3.2, right column).

## 3.4 Contrast Prescription

To overcome the inconvenient effect of energy leakage in multi-scale based contrast editing applications, we introduce a simple and efficient contrast prescription idea. Our proof-of-concept implementation allows to directly manipulate spatial selection of entire range of image scales. We define prescription-enabled editing as a manipulation in which all unmodified sub-bands are prescribed to keep their contrast values constant. As there are no assumptions made on the type of chosen multi-scale decomposition method, the contrasts in prescribed sub-bands are restored while the pyramid-like decomposition is integrated back to the output image. Adopting contrast representation from Equation 3.4, the integration is performed by simple addition of all sub-bands:

$$\log(I_{out}) = \sum_{i=1}^{N} mult_i \cdot G_i, \tag{3.5}$$

where $mult_i$ denotes per pixel contrast multiplier for band $i$ and $\cdot$ is an element-wise multiplication operator. Note that multipliers are applied to the logarithmic contrast representation, which is equal to computing a power function on the linear contrast values. In practice, this prevents too strong darkening of the image which otherwise would happen in linear space. However such an operation, in case of detail enhancement, tends to over-saturate the already well-visible details. Our prescription algorithm can counteract this scenario by selectively modulating the contrast values of over-saturated pixels. The overall contrast restoration algorithm is shown as pseudo-code in Figure 3.4.

Although we store sub-band contrast in a logarithmic ratio representation, in order to compute contrast changes during successive band manipulations, we utilize Peli's physical contrast measure, as it is a metric that can be reliably applied to complex images and by definition takes into account inter-band dependencies. We employ local sub-band physical contrast ratio to correct the contrast values affected by the cross-band energy leaking.

**Figure 3.4** – Contrast restoration algorithm

*We simultaneously integrate two pyramids using previous and current band multipliers. The integration incorporates a correction fraction for $I_1$ (output) that locally restores contrast in prescribed bands. The $I_0$ image serves as a prescribed sub-band adaptation luminance reference which is required for computing the correction factor.*

1: **ContrastRestore**(MultiScaleMultipliers *mult*, MultiScaleDecomposition $G$)
2: $N \leftarrow height(G)$ // sub-band count
3: $I_0 \leftarrow G_N$ // log(DC)
4: $I_1 \leftarrow G_N$ // log(DC)
5: **for** $i \leftarrow N-1$ **downto** 1 **do**
6: **for all** pixels **do**
7: $R \leftarrow mult_i \cdot 10^{(I_0-I_1)\cdot\beta}$ // corrected multiplier
8: $W \leftarrow 10^{|G_i|}-1$
9: $I_0 \leftarrow I_0 + G_i$
10: $I_1 \leftarrow I_1 + sign(G_i)\cdot\log(W\cdot R+1)$
11: **end for**
12: **end for**
13: $I_{out} \leftarrow I_1$

The contrast correction algorithm starts from the lowest frequency band. According to Equation 3.3, if the band-pass image is constant, the only way to change contrast is to modify the low-pass image of the same band. As we perform interactive multi-scale contrast manipulation, the aforementioned situation takes place quite often. In order to correct for this, we simultaneously compute two low-pass images $I_0$ and $I_1$ (approximated background luminance), which let us estimate for each sub-band how the contrast has changed. Initially both images are the same, so no correction is applied. However, as we move further, adding up a new sub-band components (Line 9-10 of Figure 3.4), the difference between background luminance values becomes more apparent, and directly affects the Peli contrast for prescribed bands. We modify the *mult* set to reflect the background luminance change and locally correct for the suppression or enhancement of prescribed sub-band contrast.

Given a *linear* low-pass image $I_1$ and its prescribed counterpart $I_0$, we perform pixel-wise scaling of *mult* (line 7) by $(\frac{I_0}{I_1})^\beta$ ratio, which for $\beta = 1$ corresponds to restoring Peli's contrast for a certain pixel. The $\beta$ scaling parameter provides a non-linear control over the performance of contrast restoration. As the sub-band contrast is stored in a low-pass logarithmic format, before changing the contrast value we need to convert it to an intermediate notation for which the linear band-pass signal is expressed in the nominator. For this purpose, we use the Weber fraction computed as $W = \Delta L/L = 10^{|G|}-1$. After modifying the contrast we apply simple transformation to get back to low-pass logarithmic contrast: $G = sign(G)\cdot\log(W+1)$. Corrected *mult* multipliers are stored in a separate location, so they can be used in construction of reference contrast pyramid $G$ in user's

future edits. After applying the contrast restoration formula for all sub-bands we output the image to the display.

The bottom-up approach is motivated by the fact that the most of the energy leaking is due to the large signal amplitudes of low frequency bands. This is supported by the findings in natural image statistics [GB00]: most of the image energy is focused in low-frequency bands, which is known as the power law for the amplitudes of frequencies.

## 3.5 Extension of Edge-Preserving Decompositions

In this section we discuss extending recent edge-stopping multi-scale decomposition frameworks with contrast prescription. Furthermore, we introduce supplementary extension that allows the user for counter-shading (halo editing).

### 3.5.1 Weighted Least Squares Decomposition

The contrast prescription algorithm can be implemented in WLS optimization framework in a straight forward manner. The basic idea behind the WLS based decomposition is to keep the complete frequency domain representation of each edge at only one scale. The multi-scale image decomposition is achieved by iterative application of an edge-stopping image smoothing operator, which is tuned in subsequent iterations to preserve edges of successively larger contrast. In order to convert such a representation to contrast sub-bands, we employ Equation 3.4. The smoothing operator is designed as a Poisson-class linear optimization which minimizes the energy function that penalizes the image gradients (smoothing effect) in the whole image except near strong edges (edge stopping effect). Due to the existence of high frequency edges in the band-pass image, the WLS filter requires the bands to be stored as full resolution images.

The weighting function, responsible for edge stopping behavior has been already described in Chapter 2 (Equation 2.26), and here we use its simplified form expressed as:

$$w_n(m) = \frac{1}{|L_n - L_m|^{\alpha} + \varepsilon}, \tag{3.6}$$

where $w_n$ denotes an edge weight between pixel value $L_n$ and its neighbor $L_m$ ($\alpha$ is a model parameter and $\varepsilon$ prevents division by zero). Contrast restoration mechanism (see Figure 3.4) is applied directly on the WLS multi-scale image decomposition and the resulting image is obtained by adding up the sub-bands.

### 3.5.2 Second Generation Wavelet Decomposition

Recent work by Fattal [Fat09] showed the application of second generation wavelets to edge preserving image decomposition. Here, the image is decomposed using edge avoiding wavelets (EAW) – second generation wavelets constructed with a weighting function similar to Equation 3.6. The computation of second generation wavelet decomposition is performed using the lifting scheme [Swe97][2].

Our wavelet implementation is based on a Weighted Red-Black decomposition (WRB) [URB97]. After transforming the image into wavelet representation we cannot directly use the wavelet scaling function coefficients to apply our contrast prescription algorithm. In order to recover DoG-like decomposition of the image, we compute $N$ inverse wavelet transformations. Each inverse transformation sets all scaling coefficients to zero, except the ones which describe features at scale $i$. Such an algorithm performs an edge-aware interpolation (upsampling) of selected sub-band components. The output image is a full resolution sub-band which roughly corresponds to the results obtained by the WLS framework mentioned above. As we show in Section 3.6, our GPU implementation of wavelet decomposition is very fast; the entire process takes less than 5 ms on mainstream hardware.

### 3.5.3 Perceptual Contrast Processing Framework

Mantiuk et al. [MMS06] presents a framework in which the inter-band dependencies are tightly integrated in the inhomogeneous Laplace equation and the prescription algorithm cannot be applied in the form described earlier. In this framework, especially suitable for processing HDR images, the final image is a result of least square optimization (computed using a Poisson solver) and is not reproduced by simple addition of sub-bands. In order to make our approach applicable we implement contrast restoration as a post-processing step. We use two separate decompositions to track the changes before and after manipulations, constructed by the EAW algorithm described earlier due to its efficiency. Contrast prescription is then realized using sub-band contrasts obtained from these external decompositions.

### 3.5.4 Interactive Halo Editing

The use of counter-shading to enhance the perceived contrast has been known by painters for ages [Liv02]. More recently, Krawczyk et al. [KMS07] proposed an automatic technique for improving contrast perception in digital images by modulating brightness at the edges. In our technique, we allow the user to control the halo effect (counter-shading)

[2] We refer the reader to [JO05] for a detailed discussion on second generation wavelets.

*manually*. This is implemented inside the edge-preserving decomposition by means of a minor modification in the weighting function (Equation 3.6) used by both decomposition frameworks. By modulating $\alpha$ coefficient, which is responsible for edge-stopping behavior of decomposition scheme, we can suppress or enhance halo effect near the edges. When $\alpha$ is close to 0, the weights are becoming more spatially uniform, thus the smoothing operator resembles regular Gaussian-like filter. This results in a decomposition that, in case of a local manipulation, is affected by the halo effect. For each sub-band, we define the $\alpha$ pixel-wise. To modify these coefficients we use the same approach as for updating contrast multipliers (see Section 3.6).

## 3.6 Results

In this section we present results of a comparison of prescription enabled editing with regular one on a number of images. Our proof-of-concept software[3] was tested on a mainstream PC equipped with Intel Core2 Duo 3.0Ghz CPU and NVidia GTX260 GPU. We compared the performance of decompositions presented here. In most cases the decomposition can be done off-line, as a preprocessing step before actual editing session. However, features like halo editing require recreating the sub-bands every time we modify edge weights. Consequently, we chose wavelet decomposition as our benchmark implementation since it is significantly faster than other schemes and the results we obtained are comparable. For a 1MPixel image, the forward wavelet transform coupled with generation of 8 full-resolution contrast bands takes less than 5 ms. Hence, the framework runs at interactive speeds even for large images. In case of Poisson solver, on the other hand, each iteration takes about 3 ms, depending on the amount of modification applied. Note that edits are performed iteratively on a small parts of the image. Therefore, the solver is initialized with a good quality solution, which only needs to be corrected in selected regions. On average our conjugate gradient based solver requires about 10 iterations to converge. The test software binaries are available for Windows/Linux and can be downloaded from `http://mpi-inf.mpg.de/~dpajak/prescription`.

Our implementation allows the user to intuitively manipulate the contrast multipliers. Brushing over selected areas creates a smooth amplitude mask with Gaussian-like fall-off, which is then used to update per band, per pixel multipliers (see Figure 3.5). We decided to implement brush based interface as it is still considered to be the most common tool used for manual image retouching. However, this interface can be easily extended by introducing more automated, diffusion-based segmentation as in [LFUS06]. For a novice

[3] The implementation comprises of a platform independent Java UI and native GLSL image processing library.

**Figure 3.5** – Contrast multipliers update

*The GetBandMultiplier(i) function returns* $[0,1]$ *normalized value which indicates mask scaling factor for band* $i$*. It can be either user-defined by setting up scale range sliders or computed in fully automatic manner.*

1: **UpdateMultipliers**(MultiScaleMultipliers *mult*, Image *mask*)
2: $N \leftarrow height(mult)$ // sub-band count
3: **for** $i \leftarrow 1$ **to** $N-1$ **do**
4: $B = GetBandMultiplier(i)$
5: **for all** pixels **do**
6: $mult_i \leftarrow \max(0, mult_i + mask \cdot B)$
7: **end for**
8: $mask = mask \downarrow 2$
9: **end for**

user, manipulating bands with scale range sliders can be challenging. Therefore, we include a brushing mode where band multipliers (see Figure 3.5) are computed automatically by measuring image energy[4] for current selection. This approach will always try to selectively boost the bands with smallest energy value.

### 3.6.1 Contrast prescription

We demonstrate our approach by performing a set of exemplary, yet typical, contrast editing sessions. First, we show a simple scenario, where only one band range is modified and then we stage more complex manipulation to show how bands interact with each other.

In Figures 3.1 and 3.7, we perform single modification of fine image details using WRB Wavelet decomposition scheme with parameters $\alpha = 0.8$ and $\beta = 1.0$. In both figures, we used the same band range multiplier for prescribed and non-prescribed operation. Due to the energy leakage, features existing across multiple scales are over saturated and by comparing against the source image we see that their contrast enhancement is spatially inconsistent. Prescription counteracts these situations and allows for more uniform and controllable contrast modification. Note that it is cumbersome to achieve such a result with a series of separate edits since the restoration is spatially local and highly non-linear.

Figure 3.8 shows an opposite scenario, where user scales low frequency bands. The modification causes the loss of visibility of trees and plant pot details. Prescription of contrasts in upper bands allows to achieve both goals, increasing the global contrast and maintaining the detail visibility.

[4] Modulo of a gradient for gradient domain frameworks and absolute amplitude of band-pass contrast for multi-scale decompositions.


**Figure 3.6** – Contrast editing session in WLS decomposition framework

*Contrast prescription prevents the loss of details in unmodified sub-bands, as a result, previously modified fine details are preserved.*


**Figure 3.7** – An example of contrast enhancement using WRB wavelets

*We used* $\alpha = 0.8$ *(Equation 3.6) and* $\beta = 1.0$ *(Figure 3.4) in the decomposition. Boosting fine/medium scales in the ceiling area causes saturation of some image features and results in unnatural appearance. Despite the extreme boost, contrast restoration allows to regain natural look of modified area and get the requested detail enhancement.*

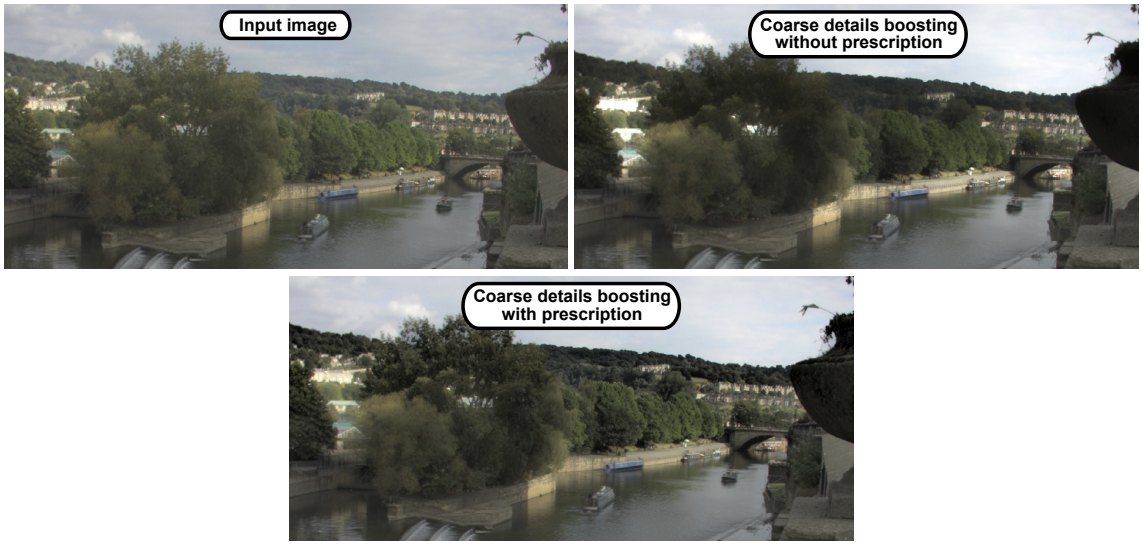


**Figure 3.8** – Regular and prescription-enabled image editing in WLS framework

*Amplification of coarse features attenuates contrasts in higher bands. Contrast restoration algorithm successfully recovers fine details and boosts coarse image features at the same time.*

In Figure 3.6, the image is initially modified by enhancing fine details around the plant area. Next, we boost low frequency bands, which reduces the visibility of previous edits. Also, high signal amplitudes of low frequency content exposes the energy leakage issue, resulting in an over-saturated image and decreased perception of unmodified contrasts bands. Contrast prescription visibly restores the details and reduces the saturation while still allowing for large enhancement of low frequency contrasts.

Finally, we illustrate the complete editing session result, Figure 3.10, where we manipulate contrasts in order to transfer the style of a professional photographer to a plain picture of the same location. Despite the obvious difference in source material we managed to properly reflect the style using only local, prescription-enabled contrast modifications.

### 3.6.2 Interactive Halo Editing

As described in Section 3.5.4, we enhanced each of the implemented decomposition frameworks to allow interactive halo manipulation. Figure 3.9 illustrates simple, low frequency halo suppression case. Although the local modification of edge-stopping filter behavior usually requires repeating the decomposition, we show that an efficient hardware implementation can deliver an interactive solution even in case of Poisson solver based frameworks.

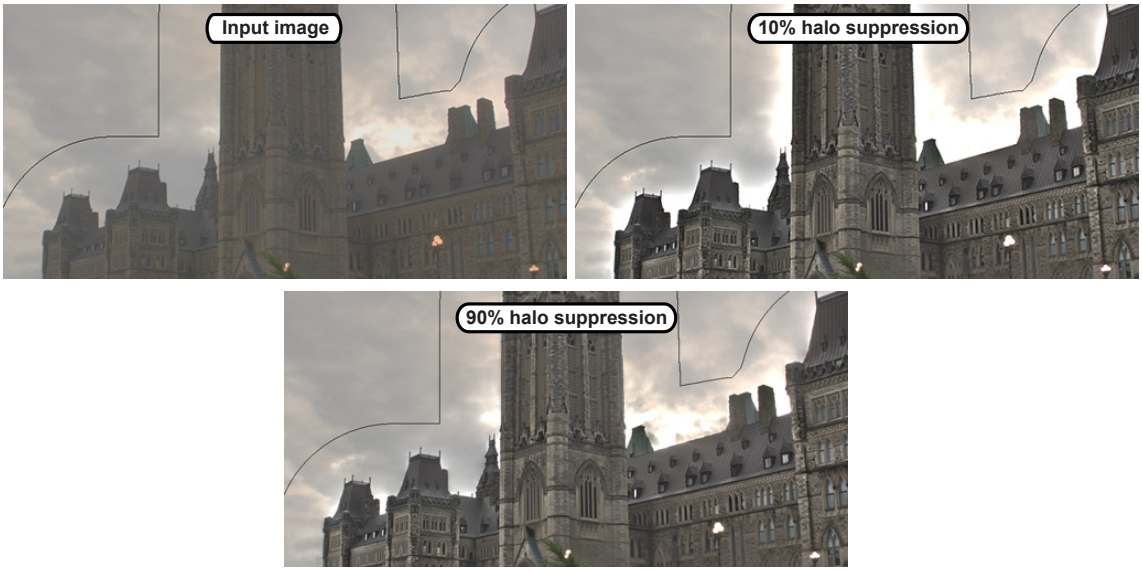


**Figure 3.9** – Halo editing example using a Poisson solver [MMS06]

*The original castle scene is modified by elevating high/medium frequency bands. To artificially modulate contrast along the edges, e.g. for better decomposition of foreground from background, one might locally allow the halo effect to be visible.*

## 3.7 Conclusions and Future Work

In this work, we analyzed the properties of multi-scale image representations used in interactive image editing applications. Direct consequence of Gaussian-like filters commonly used to construct such representations (including edge-aware decompositions), is the effect that we call "energy leaking". When the user modifies one sub-band, part of the "energy" of this modification in effect leaks out to the other sub-bands in successive manipulations. This can affect the perception of already edited parts of the image and leads to non-intuitive, iterative way of image editing. Moreover, change in one band can result in oversaturation of another band (also possibly previously edited), e.g. when user boosts overall contrast, the tiny details might be lost. To overcome those limitations inherently imposed by all existing multi-scale image editing frameworks, we propose the concept of contrast prescription, where once edited parts of the edited image retain their prescribed values. The aim is to restore the visibility of each sub-band contrast and limit the effects of cross-band energy leakage. Consequently, reduced number of decomposition related artifacts, allows for more intuitive and controllable multi-scale contrast manipulation. To illustrate the concept, we show simple, but efficient and interactive extension of three state-of-the-art multi-scale frameworks.

So far, we assumed editing using an ordinary (LDR) display device – in this case the illumination stayed almost constant and the effect of image editing on the user's visual

**Figure 3.10** – A practical demonstration of prescription enabled contrast editing

*We modified the source image(left) to reflect the style and feeling of pictures taken by Ansel Adams (right). As the artist style was based mostly on manual dodging and burning we had to perform extreme band manipulations that applied without prescription would result in heavy artifacts. The entire session was about 8 minutes long and included only contrast manipulations on WLS based decomposition (center).*

adaptation was very subtle. However, when editing on an *HDR display* device, the change of user's adaptation due to the display luminance is not negligible any more and it can significantly bias user's perception. Modelling of apparent contrast is required to compensate for this effect which suggests a possible extension to this work.

# Contrast-enhanced Compression of Depth and Motion Video Stream

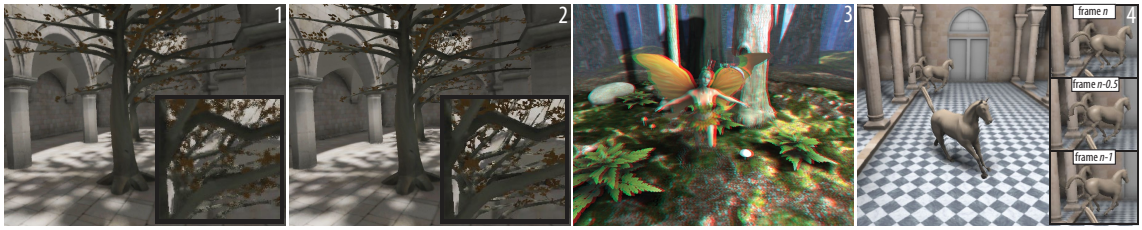


**Figure 4.1** – Possible applications of our enhanced streaming architecture

*Remote rendering allows navigating in complex scenes even on weak client hardware. But not only final images are of interest on the client side, auxiliary information like depth or motion become increasingly attractive in this context for various purposes. Examples include spatio-temporal upsampling (1, 2), 3D stereo rendering (3), or frame extrapolation (4). Standard encoders (H.264 in image 1) are currently not always well-adapted to such streams and our contribution is a novel method to efficiently encode and decode augmented video streams with high-quality (compare insets in image 1 and 2).*

In this chapter we focus on efficient compression and streaming of frames rendered from a dynamic 3D model. Remote rendering and on-the-fly streaming become increasingly attractive for interactive applications. Data is kept confidential and only images are sent to the client. Even if the client's hardware resources are modest, the user can interact with state-of-the-art rendering applications executed on the server. Our solution focuses on augmented video information, e.g., by depth, which is key to increase robustness with respect to data loss, image reconstruction, and is an important feature for stereo vision and other client-side applications. Two major challenges arise in such a setup. First, the server workload has to be controlled to support many clients, second the data transfer needs to be efficient. Consequently, our contributions are twofold. First, we reduce the server-based computations by making use of sparse sampling and temporal consistency to avoid expensive pixel evaluations. Second, our data-transfer solution takes limited bandwidths into account, is robust to information loss, and compression and decompression are efficient enough to support real-time interaction. Our key insight is to tailor our method explicitly for rendered 3D content and shift some computations on client GPUs, to better balance the server/client workload. Our framework is progressive, scalable, and allows us to stream augmented high-resolution (e.g., HD-ready) frames with small bandwidth on standard

hardware.

## 4.1 Introduction

Server-client platforms that enable remote 3D graphics interaction are of high importance in the age of mobility and constant hardware change. By relegating rendering to powerful servers and streaming of the resulting frames, the requirements on the client side can be reduced to standard web browsing and video playing abilities.

Server-client graphics platforms have many more advantages, e.g., potentially huge data sets remain on the server which simplifies maintenance, improves security, and enables consistent updates in collaborative work scenarios. Consequently, numerous applications can benefit from this setup. Examples include medical 3D visualization, industrial and architectural design, at the spot repair of complex devices (e.g., ships and aircrafts), 3D navigation and tourism, and in particular online entertainment and gaming. In the latter case, players subscribe to an instant access of a large variety of games. Server-side execution makes client-hardware upgrades, installation and software updates mostly unnecessary. These advantages are attractive for customers, as well as game developers that benefit from reduced marketing / distribution costs and piracy risks. Not surprisingly, a number of companies such as OnLive, OTOY, Gaikai are committed to provide technology for game streaming.

The key problems of such an approach are the server workload that limits scalability with respect to the number of clients and bandwidth. Particularly, standard compression schemes are often expensive and introduce a long latency. HD-ready-resolution frames need approximately 3 Megabits per second for good quality encoding using traditional streaming with H.264. However, usage of MPEG B-frames introduces delay which is unacceptable for real-time applications, like games. Low-delay streaming requires more data transfer (e.g., OnLive recommends 5 Megabits or more for HD-ready), but effects such as 3D stereo, or high frame rates, would quickly lead to an excessive bandwidth.

Our insight is that transfer costs can be reduced by augmenting the stream with supplementary information, such as depth or motion. These *attribute buffers* are very cheap to compute on the server, and allow many applications when transferred to the client, including warped 3D stereo, spatio-temporal upsampling (super-resolution), and frame extrapolation. All of these benefit from the client-side execution — 3D (acceptable disparity depends on eye distance and device size), resolution and framerate can be optimized for the client's equipment or preferences. The only problem is that such applications require high precision attribute buffers which makes standard compression schemes difficult

to apply. Our approach addresses this issue by linking rendering and compression, while existing streaming systems (e.g., Reality Server, OnLive) benefit little from the fact that a 3D model is underlying the rendering.

While we follow the (usually valid) assumption that the client machine is much less powerful than the server, a pure video-streaming scenario makes almost no use of the client's computational capabilities. Instead we want to exploit this resource. To transfer high-precision auxiliary data (e.g., depth) to the client, our solution uses an efficient edge encoding and a new fast diffusion process. Consequently, on the server, costly raytracing or per-pixel shaders are only applied to low resolution frames, that are streamed to the client where the corresponding high-resolution frames are reconstructed, hereby lowering server workload and bandwidth.

Precisely, our contributions are:

- **Compression** – an efficient scheme not relying on future frames and a novel inter frame prediction;
- **Precision** – high-quality discontinuity edges to enable spatio-temporal upsampling;
- **Decompression** – a novel GPU-adopted diffusion scheme and decoding;
- **Transfer** – a solution to adapt bandwidth.

We illustrate the advantages of our motion-augmented video streaming with several client-side applications (e.g., stereo rendering, temporal super-resolution).

## 4.2 Related Work

Only few successful attempts exist that couple 3D rendering, efficient data compression and streaming. Cohen-Or et al. [COMF99] and Levoy. [Lev95] enhance client rendering by streaming residuals (difference between high-quality-server and low-quality-client frames). The server workload increases because residuals require both, the client and server-side rendering. Also, the client needs to receive a 3D scene description. In the limit, one could stream only graphics API calls [NDS*08], but powerful clients are needed and server-side rendering is impossible.

We execute scene-related costly computations on the server. The client's cost depends only on the image resolution. To reduce server costs, we build upon amortized rendering techniques [SaLY*08, YSL08, HEMS10]. Usually on a single machine, we employ them in a streaming context and let the client reconstruct the image.

We can also exploit knowledge concerning the final compressed video. In the Render2MPEG

framework [HKMS08], details that would be removed during compression are not rendered. Similarly, we can use a low bandwidth or limited client display capabilities to reduce the server load.

One major contribution of our work is a novel depth encoding. In particular, free viewpoint video (FVV) and 3D television [KAF*07] (3DTV) require the so-called multiview video (MVV) representations, which often involve the depth data to warp the original video stream to nearby virtual camera positions. Because video codecs such as H.264 are optimized for image statistics and human perception, depth compression requires specialized solutions [MMS*09]. Many approaches smooth depth values in order to increase compression performance [Feh04] at the cost of crucial precision. Lossless depth compression via a run-length like encoding [JWB04], or by integrating an underlying 3D model [EWG99] might not easily meet bandwidth constraints. Consequently, we aim at high accuracy, but allow a tradeoff between quality and compression to meet bandwidth limitations.

Precision is most crucial at depth discontinuities [MMS*09]. Edge-preserving filtering changes depth values, but lets the encoder focus more on discontinuities [PJO*09]. Also, region-of-interest specifications and depth-value redistribution [KbCTS01] can improve quality. Nonetheless, depth discontinuities are only handled implicitly in the encoding step which can lead to significant divergence [MMS*09].

Merkle et al. [MMS*09] introduced a different depth-optimized encoding for adaptive pixel blocks (possibly separated by a single linear edge) and assign a constant or linear depth approximation (if present for both edge sides). Fitting the edges and constructing the representation is costly, but leads to a quality leap. We avoid explicit optimizations and show that we achieve higher quality with higher encoding efficiency. We also introduce a temporal prediction which is very effective for animated sequences.

The importance of depth discontinuities is also illustrated by approaches that perform joint color / depth compression [MD08] because of the often strong correlation. In similar spirit, we will rely on depth information to perform spatio-temporal upsampling to reduce the server workload, but instead of a semi-automatic solution [MD08], we need an automatic and sufficiently efficient online solution.

Image encoding with H.264 is computationally expensive. Half the time is used on neighborhood matching and motion estimation [CBPZ04] that can be directly recovered from 3D [FE09] and enable higher compression [WKC94]. In our solution, we rely on motion vectors to enable spatio-temporal upsampling in dynamic scenes. Their high redundancy [MF98] allows for an efficient encoding.

Our solution benefits from the fact that image information is strongly correlated with discontinuities [Eld99]. Edges can be beneficial for compression [GWW*08], but the construction and reconstruction steps are usually costly. The locality of our approach makes

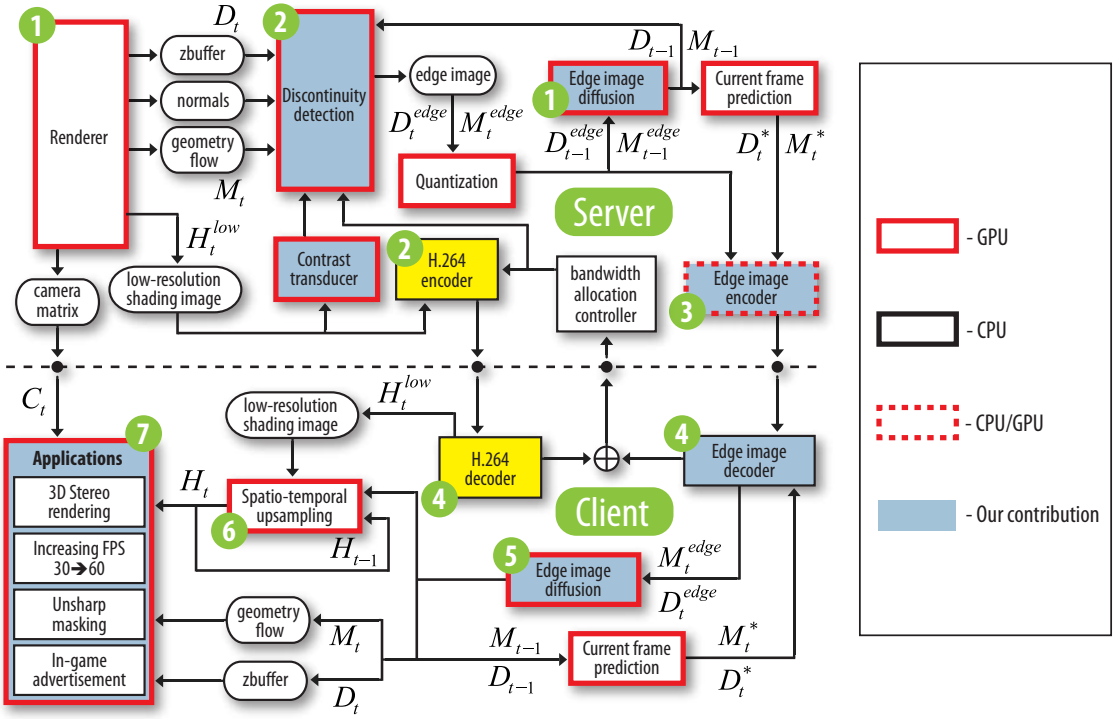


**Figure 4.2** – Outline of the proposed streaming architecture

*Green markers indicate the computation order within a frame cycle.*

it efficient and ready for an online rendering context.

## 4.3 Augmented Streaming Framework – An Overview

Our algorithm consists of several components, which are depicted in Figure 4.2. On the server side, we render a jittered low-res. version $H_t^{low}$ of the current frame $H_t$. Avoiding the production of a full-res. frame reduces the server workload and saves bandwidth to be used for our additional streaming data. The image $H_t^{low}$ is encoded using a state-of-the-art H.264 video encoder and sent to the client who constructs a high-res. version $H_t$ based on the previously reconstructed high-res. frame $H_{t-1}$. Key for this upsampling are high-res. depth information of the current frame $D_t$ and the motion flow $M_t$ between the current and previous frame. Both are cheap to compute on the server [SaLY*08].

To efficiently transfer depth $D_t$ and motion $M_t$ to the client, we rely on a customized compression scheme. We detect discontinuities that define edges on the server (Section 4.3.1). The edges and their depth and motion values $(D_t^{edge}, M_t^{edge})$ are encoded rapidly using previous information $(D_{t-1}, M_{t-1})$, then sent to the client. The client reconstructs a high-res. depth $D_t$ and motion flow image $M_t$ by diffusing the sparse edge information (Sec-

tion 4.3.2), hereby exploiting smoothness of the signal between discontinuities. The client computes the final image $H_t$ using the low-res. image $H_t^{low}$, the depth, and motion flow (Section 4.3.3).

Our compression scheme outperforms state-of-the-art encoders (Section 4.6) and enables a high-quality spatio-temporal reconstruction. We propose a bandwidth control based on a perceptual metric (Section 4.3.4) and provide implementation details of our low-level encoding process (Section 4.4). The relatively accurate depth $D_t$ and motion $M_t$ data, enable many applications on the client side that were difficult to achieve with previous encoding strategies (Section 4.5).

### 4.3.1 Server-Side Preparation

It is very cheap to produce high-res. *attribute buffers* (depth, surface normal, motion, and texture) which is typically done for deferred shading. The main server cost stems from producing the final shaded image, especially when using raytracing. Using a lower resolution image $H_t^{low}$ leads to a performance gain. We then stream $H_t^{low}$ via H.264.

A H.264 codec is not optimal for high-res. depth $D_t$ and motion $M_t$ whose precision is crucial for the client-side upsampling that transforms $H_t^{low}$ into a high-res. high-quality output $H_t$. Also other applications (Section 4.3.3) would suffer. In contrast to *standard* image information, depth or motion have little visual meaning and traditional video-coding (e.g., MPEG) is perceptually-tuned. Instead, discontinuities in the attribute buffers should be preserved, while precision can be lowered for the remaining image. E.g., frame extrapolation with a low-quality motion flow will be most perceivable at contrast-discontinuities or region boundaries [SLW*08]. We will show that $D_t$ and $M_t$ can be compressed and streamed efficiently by relying on a customized edge-based compression and client-side reconstruction via diffusion. The result closely resembles the original on edges and smoothly interpolates between them.

Edges are detected from all attribute buffers. One could use a gradient-based edge detector, but it often fails in practice. E.g., a flat surfaces seen at a grazing angle is detected as discontinuity. Instead, we rely on a Laplace operator. For performance reasons, we detect edges from depth and motion-flow buffers only. A weighted sum of responses from all buffers yields pixel significance. Predefined threshold selects significant pixels, the so-called *edge samples* (in Section 4.3.4, we show how to improve upon this ad-hoc threshold). To avoid missing low frequency changes, we also uniformly add one sample every $32 \times 32$ pixels.

We do not define curves explicitly and only select edge samples, but a Laplacian detects edge structures (Figure 4.4) well. Further, it always extracts two-pixel-wide edges. Con-

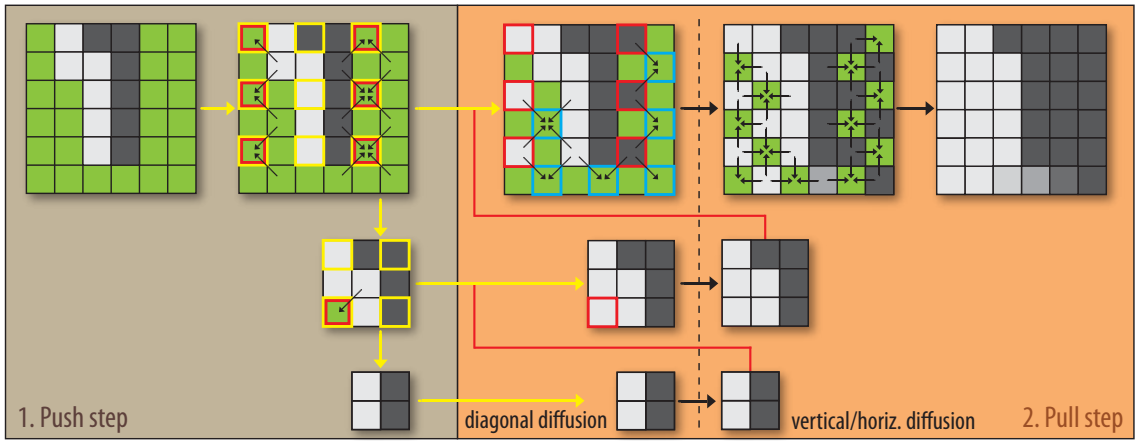


**Figure 4.3** – Custom fast diffusion scheme

*Samples stored on both sides of an edge are spread over the image in two stages. 1) Push phase: downsample image (yellow rectangles). Try to fill empty (green) samples by averaging diagonal neighbours. 2) Pull phase: coarse level images fill in holes in finer scales. First, transfer copied values (previously yellow, now red rectangles). Fill empty pixels (blue rectangles) by averaging diagonal, then adjacent samples.*

sequently, we keep two values, one for each side along the discontinuity. Only in this setting, a diffusion process can reproduce the discontinuity. Besides depth and motion, we store the sign of the Laplacian detector (zero-crossing of 2nd derivative) because — almost always — neighboring pixels with the same sign also lie on the same surface. This information will be used to improve the compression efficiency.

### 4.3.2 Client-Side Reconstruction via Diffusion

The client receives a low-res. shading image, and edge-based representation including values for depth and motion. In this section, we propose a fast diffusion scheme used to reconstruct high-res. depth and motion information.

The basic idea of our method is a push-pull mechanism [GGSC96]. The goal of the *push step* (Figure 4.3, grey) is to fill holes by reducing resolution. We create a pyramid-like image representation by successive downsampling:

$$D(x,y) = \begin{cases} I(2x,2y) & ,w(2x,2y) > 0 \\ \frac{\sum_{x',y' \in N_{2x,2y}} w(x',y') I(x',y')}{\sum_{x',y' \in N_{2x,2y}} w(x',y')} & ,w(2x,2y) = 0 \end{cases} \tag{4.1}$$

where $N_{x,y} = \{(x-1,y-1),(x-1,y+1),(x+1,y+1),(x+1,y-1)\}$ stands for a local neighborhood, $I$ is the edge-based image and $w$ contains binary weights which are set to 1.0 for edge samples, and 0.0 for empty pixels. This operation extends the edge samples by one pixel in each level and holes are quickly filled. Due to the uniform edge samples (one sam-

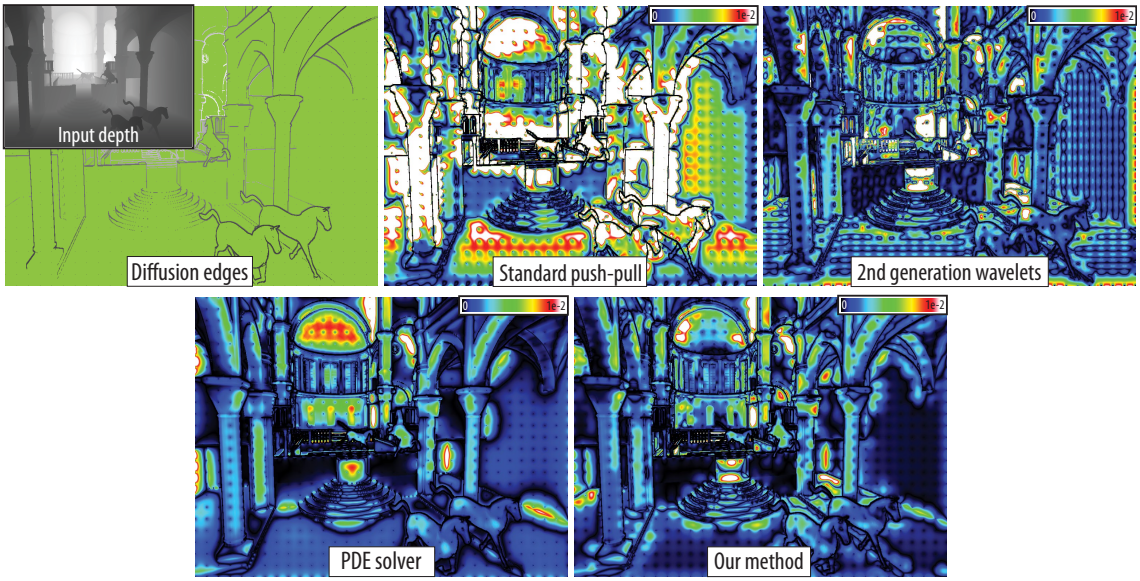


**Figure 4.4** – Depth buffer edge-diffusion error visualization

*Input edge image (left) has been processed with selected "inpainting" algorithms. Our method provides quality comparable to PDE solvers, yet it is much less computationally intensive. Also, it has the smallest relative error of approximation, which is particularly important for our problem. Measured PSNR: push-pull: 36.45dB, 2nd generation wavelets: 48.47dB, PDE: 52.42dB, our method: 52.49dB.*

ple per $32 \times 32$ pixels), five levels of the pyramidal downsampling are sufficient to entirely fill the top image in the pyramid.

After the push step, we use a *pull* operation (Figure 4.3, orange) to propagate the filled hole back to the high-res. image. In a top-down manner, coarse-level samples are "pulled" back to their corresponding position in the finer level. We do not overwrite existing fine-level values as these were defined during the push step and are thus based on a more-precise higher-resolution image. The push step kept even pixel values, consequently, during the pull step, we only transfer values from these even pixels (red squares). The remaining pixels of the current pyramid level are filled in local diffusion steps (we apply a diagonal (cyan) then axis-aligned diffusion which performed best in our tests).

To further improve the smoothness of the output surface, similar to multi-grid methods, one can apply a pre or post-smoothing step on each level in the *pull* stage. A single Jacobi iteration reduces local diffusion errors, but also slightly increases the average approximation error. Compared to standard push-pull, our diffusion has reduced anisotropy (due to *quincunx lattice* sampling scheme) and respects edges. Additionally, we process screen-space linear input and therefore diffuse in a perspective correct way.

### 4.3.3 Client-Side Spatio-Temporal Upsampling

Having all data available on the client, the final high-res. shaded image needs to be constructed. One could simply scale the low-res. shaded image $H_t^{low}$ to the screen resolution, but then the quality would be poor. Instead, one can significantly improve quality when performing a spatio-temporal reconstruction (following [HEMS10]).

To reconstruct the final image, the client relies on the reconstructed attribute buffers of the current frame: depth $D_t$ and motion flow $M_t$. Using $M_t$, pixels in the current frame can be projected back into the previous high-res. frame $H_{t-1}$ which is still present on the client. As in [HEMS10], a bilateral weighting scheme (screen position, depth, and distance to the samples in $H_t^{low}$) attributes weights to the pixels in the temporal and spatial neighborhood in $H_{t-1}$ and $H_t^{low}$. The weighted sum defines the current frame $H_t$.

Disocclusions (pixels in $H_t$ whose corresponding sample was either outside the viewing frustum or occluded in $H_{t-1}$) need to be considered in the weighting scheme. In this case, no reliable information exists in $H_{t-1}$ and non-zero weights are only attributed to pixels in the current frame $H_t^{low}$.

The complete spatio-temporal upsampling is more complex than described, e.g., illumination gradients are used to treat dynamic lighting. The interested reader is referred to [HEMS10]. In particular, the server creates jittered frames that ensure a convergence to a high-quality rendering for static elements. The major difference is that we do not use normals to reduce bandwidth. We counteract this restriction with a twice bigger spatial reconstruction kernel for higher accuracy. This larger support is possible because of our streaming context: the client avoids the scene rendering and offers many unused resources.

### 4.3.4 Scalability

In an online rendering context, it is important to adapt the bandwidth dynamically. One could adapt the resolution of the rendered video stream effectively, hereby changing the upsampling ratio, or reduce the frame rate by making use of temporal upsampling (see Figure 4.1) on the client. While the previous measures have a strong impact on the bandwidth, but also quality, one can also control the bandwidth/quality tradeoff on a finer level in the edge encoder.

When many edges arise from the discontinuity detection, our encoding becomes suboptimal. Fortunately, not all edges are needed. “Edge importance” strongly relates to the perceptual contrast in image $H_t$. The more cluttered an image region becomes, the less precision is needed due to perceptual masking effects. Hence, we would like to weight geometric edges according to “visual importance”. However, since this computation is

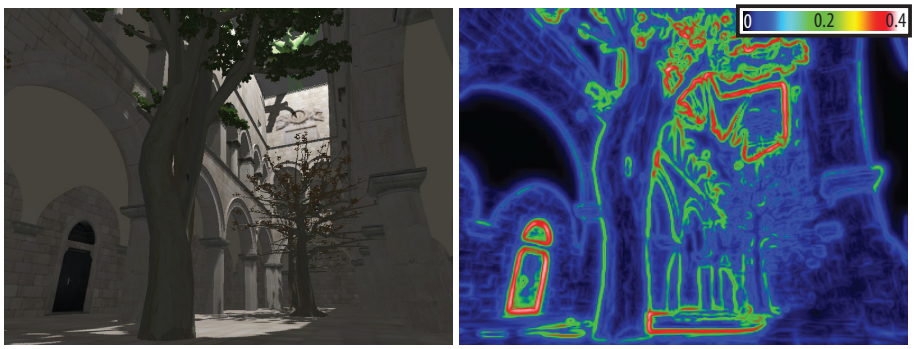


**Figure 4.5** – A complex frame and the visual significance of its edges

*We denote dark blue edges as imperceivable and red edges as clearly visible. To meet bandwidth constraints, we favor important geometric edges. Note the small edge importance in the tree, which is dominated by visual masking.*

executed on the server, we want to avoid high costs, therefore we focus on the available low-res. rendering $H_t^{low}$. Consequently, masking thresholds are computed via the GPU only for coarse scale edges on low-frequency bands of a Gaussian pyramid.

Precisely, we convert the low-dynamic-range image $H_t^{low}$ to linear luminance (we ignore isoluminant chroma edges) and build a Gaussian pyramid. Using a Sobel filter, we compute the magnitude $c_l(x)$ of the gradients in each level. These gradients are analyzed for visual masking with a contrast transducer borrowed from JPEG2K [ZDL00]:

$$R_l(x) = \frac{|c_l(x)|^{0.6}}{1.0 + \sum_{i \in \mathcal{N}_8(x)} |c_l(i)|^{0.2}}, \tag{4.2}$$

where $\mathcal{N}_8(x)$ is the eight-pixel neighborhood around pixel $x$ in each pyramid level. The apparent contrast map $R(x)$ is the sum of the band responses $R_l(x)$ (Figure 4.5). The final edge importance that guides the edge-sample selection, is computed by multiplying $R(x)$ with the weights derived from the attribute buffers. Further, we blend the edge weights with the motion-compensated edge weights of the previous frame to enforce temporal coherence and reduce edge flickering.

By exploiting the new edge weighting, we can achieve a *constant bit rate control* (CBR) by increasing/decreasing the number of edge pixels. To this extent, we vary the edge thresholds by a rate control factor, which exponentially converges to the desired target bit rate. Furthermore, although not needed in our tests, one can increase/decrease the edge-data quantization (see Section 4.4).

## 4.4 Stream Compression

We showed how the edge-based representation allows us to transfer high-quality high-res. images, but, for this transfer to be efficient, we need to encode the data properly. This section explains our efficient compression technique. We pursue two goals: a fast encoding and a good compression.

We encode two elements, first, a binary image defining the existence of edge samples (1=edge sample, 0=empty); second, the edge-sample values that are stored in these pixels. The binary image compresses well and defines a topological layout. Edge-sample values (such as depth, motion-flow or edge laplacian "sign") are then stored in scanline order and the binary image allows us to attribute the values to their original image locations.

### 4.4.1 Binary Image encoding

One of the most effective schemes for lossless binary image encoding is JBIG2 [ITU99]. It employs adaptive binary arithmetic coding [WNC87] to reduce the size of the resulting bit-stream. Unlike Huffman coding, arithmetic coders can output symbols occupying less than 1 bit on average. Such high compression ratios are possible by modeling probabilities (symbol frequency tables) based on the spatial neighborhood of the encoded characters. The adaptivity of the scheme ensures that more frequent symbols will require less bits to encode. This *higher-order* modelling is especially suitable for compression of images showing regularly structured elements (e.g., fonts, simple shapes). Usually the spatial neighborhood consists of previously encoded pixels (assuming scanline traversal) – the values beyond the encoding position will not be available to the client when performing the decompression. For binary edge images, this restriction resulted in lower efficiency. Further, JBIG2 was designed for still image compression, whereas we want to exploit inter-frame coherence and temporal prediction. Therefore, we propose a different approach.

To improve the intra-frame compression, we split the binary image into four interleaved sets of pixels. During the first pass, we predict values using only previous pixels, but from the second pass on, pixels extracted from previous runs are available. For any compression position, we can thus rely on pixels that lie in the "compression future" (Figure 4.6, pass 1–3). Intuitively, we can better predict where edges lie in the image. For edge-only images, samples from different passes show a high spatial correlation (Figure 4.6, pass 3). The impact on the compression efficiency is significant. In practice, the first pass uses approx. 50% of the file size, while all three following passes are squeezed in the remaining 50%. In contrast to a standard JBIG2 encoding, our intra-frame scheme generates 25% smaller frames on average.

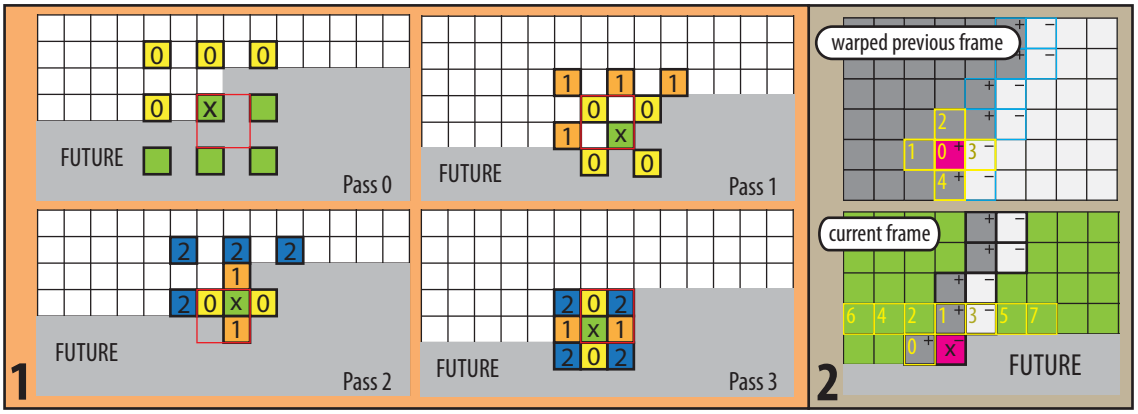


**Figure 4.6** – Compression prediction schemes

*1) Spatio-temporal contexts for edge-image compression. To compute probabilities for sample 'x', a 9-neighbor spatio-temporal context is used, choosing from current pass, previous pass (numbered pixels) or the motion-compensated previous frame (green pixels). Combining values from these various origins keeps the prediction rates high. 2) Spatio-temporal value prediction. We select two potential predictors for current sample (purple cell in current frame): one from current and the other originating from warped previous frame. Candidates are chosen from a fixed neighborhood set (yellow cells). The first neighbor that matches the current sample's sign is selected as the "best" predictor.*

To exploit inter-frame coherence, previous-frame samples are added to the current sample encoding context by warping the previous frame into the current view. Any warping strategy would work, but we use the fast solution we also employ for stereo rendering (Section 4.5). We can then access values of the previous frame during en- and decoding (green pixels in Figure 4.6). Depending on the quality of the warped frame, this context extension increases compression ratios between 1.2 and 7 (on average 2.5).

### 4.4.2 Edge-Sample Values

To encode the edge-sample values (depth, motion vectors), we make use of the binary image. Basically, we group all edge-sample values in scanline order and compress this number stream. To reduce data entropy, we first apply a spatio-temporal predictor. The resulting residual stream is entropy coded before being sent.

Precisely, during the prediction step (Figure 4.6, 2), we find a value that best matches the currently encoded sample. We assume that such a value will exists in a close spatial neighborhood and most likely will belong to the same geometric surface (same sign of the Laplacian operator). The residual is then computed by subtracting the match from the current pixel value. As depth discontinuities are highly correlated with motion discontinuities the best predictor for depth values will also be a good candidate for motion prediction.

In order to choose the best predictor we offer ourselves two choices, the first sample with the same Laplacian sign on the edge in the current or in the previous frame. In each step, we keep track of the used strategy in a table containing all possible $3 \times 3$-binary edge configurations. The strategy to choose a new sample is based on the corresponding entry, if current frames proved more successful, we apply this strategy, otherwise we use the previous frame. Afterwards we verify whether this choice was effectively the better one. We cannot do this before because we need a deterministic scheme that allows us to execute the same steps on the client for decompression. Based on the verification, we update our table entry which will affect the decision for the next similar context. In this way we probabilistically train a good predictor for each given context.

To reduce the range of the residual values, one can perform additional quantization which results in increased compression ratios and only slightly reduced accuracy. In our implementation, a quantization factor of 2 generates frames with approx. 90% of zero residual coefficients.

In order to capture low-frequency value changes, we added an additional edge-sample value every $32 \times 32$ pixels. These regular samples define a low resolution image that can be compressed separately. This way, we better preserve the edge-like nature in the remaining image which is exploited by our encoding algorithm. In practice, the low resolution image is encoded using the Paeth predictor [Pae91] followed by entropy coding.

### 4.4.3 Motion Flow Quantization

For static objects, motion vectors are only a consequence of camera movement, which is known by the client. Hence, we encode only the non-camera movement of dynamic object pixels. We further quantize 2D motion vectors non-linearly by applying a power function (0.5), which favors slower motion, as these movements are most crucial for the upsampling process. Only fast moving dynamic objects might exhibit some small artifacts which are usually invisible because our detail perception on moving objects is much lower than for static elements. Consequently, the motion flow encoding proved less critical. Its spatial and temporal coherence leads to a good compression.

## 4.5 Applications

Besides the discussed spatio-temporal upsampling of lower resolution frames, the augmented depth/motion information allows for various other applications on the client-side. We will briefly introduce and discuss a few important ones.

### 4.5.1 3D Stereo Rendering

Knowing the camera and depth is enough to compute 3D stereo via image-based warping. We use the technique proposed in [DRE*10], where a grid is overlayed with the depth buffer. The grid's vertices are snapped to the pixel corners between foreground and background depth-buffer pixels to reduce artifacts. We then transform this grid to simulate a left and right eye view. Disoccluded pixels can optionally be blurred to reduce the visibility of artifacts. The procedure runs entirely on the GPU and takes less than 2 *msec* (nVidia GTX 280) for the grid warping with a resolution of $320 \times 180$ vertices which is sufficient for an HD-ready resolution. In Figure 4.1, image 3, the detailed leaves and the fly in the foreground are nicely warped, which shows the importance of precise depth edges.

### 4.5.2 Temporal Upsampling

We can also interpolate or extrapolate frames in time (Figure 4.1). Such an approach has been proposed in [DER*10]. The current frame is warped to the future by extrapolating the motion vectors and using the depth for occlusion. Of course, non-linear motion and motion discontinuities can result in overshoots. Such scheme is most sensible for high frame rates above 30 Hz to reduce LCD-display hold-type blur [DER*10]. Nevertheless, it is useful if frames are dropped during transmission.

### 4.5.3 Advertisement and Highlighting

The depth buffer can be used for many interesting effects to highlight elements, e.g., Unsharp masking [LCD06]. Although such an operation could be employed on the server side, for sports events, where many clients share the same data stream, high-lighting can be done locally, thus allowing an observer to track their favorite player or adjust display settings (e.g., screen-space ambient occlusion is part of many display drivers, but not everybody appreciates and activates the mode). Another example is online advertisement. The supplementary streams make it possible to seamlessly integrate new content that can directly target the interacting user.

## 4.6 Results

We developed a proof-of-concept OpenGL prototype, computing all steps of our streaming system, however simulated without network component. We run the full application (Server and Client) on a desktop system equipped with an Intel Core 2 Quad Processor

| Scene | Resolution | Server | | | | | | | Client | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $T_{ren}$ | $T_{h264}$ | $T_{ren}^{low}$ | $T_{pre}$ | $T_{enc}$ | *Fps* | *Speedup* | $T_{dec}$ | $T_{dif}$ | $T_{up}$ |
| SIBENIK | SVGA | 57 | 32.3 | 8.6 | 3.8 | 26.6 | 25.6 | ×2.3 | 19.0 | 1.3 | 1.2 |
| | HD 720p | 106 | 46.0 | 14.5 | 6.2 | 26.8 | 21.1 | ×3.1 | 20.3 | 1.5 | 2.3 |
| | HD 1080p | 251 | 73.7 | 29.1 | 13.9 | 47.1 | 11.1 | ×3.6 | 38.4 | 3.4 | 4.8 |
| SPONZA | SVGA | 63 | 32.4 | 9.1 | 3.9 | 29.7 | 23.4 | ×2.3 | 23.4 | 1.5 | 1.2 |
| | HD 720p | 120 | 45.0 | 15.1 | 6.9 | 35.2 | 17.5 | ×2.9 | 28.9 | 2.1 | 2.2 |
| | HD 1080p | 271 | 76.4 | 30.7 | 14.4 | 57.0 | 9.8 | ×3.4 | 48.0 | 3.9 | 4.7 |
| FAIRY | SVGA | 32 | 31.0 | 7.2 | 3.8 | 24.2 | 28.4 | ×1.8 | 16.0 | 1.3 | 1.1 |
| | HD 720p | 51 | 44.4 | 10.6 | 7.3 | 32.3 | 19.9 | ×1.9 | 24.6 | 2.6 | 2.1 |
| | HD 1080p | 107 | 72.2 | 16.7 | 15.7 | 54.6 | 11.5 | ×2.1 | 46.7 | 2.8 | 4.7 |

**Table 4.1** – Server and Client timings

*Timings in msec for different target resolutions:* 800×600 *(SVGA),* 1280×720 *(HD 720p),* 1920×1080 *(HD 1080p). Rendering costs are dominated by frame size and all frames are* 4× *upsampled. For comparison the costs for full-res. rendering ($T_{ren}$) and video-encoding with H.264 ($T_{h264}$) are provided together with the savings by our method in column (Speedup). Our server-side timings comprise: low-res. rendering (deferred shading) ($T_{ren}^{low}$), the time for previous frame depth/motion warping plus edge diffusion (simulating client steps) plus all other GPU steps involved in the pipeline ($T_{pre}$) and, finally, the arithmetic and H.264 low-res. encoding on the CPU ($T_{enc}$). On the client-side we perform: low-res. H.264 stream and edge decoding including depth/motion frame prediction ($T_{dec}$), and edge diffusion ($T_{dif}$) followed by upsampling the low-res. video frame ($T_{up}$).*

Q9550 and a NVIDIA Geforce GTX 280 graphics card and, separately, the client-side operates on a notebook computer with an NVIDIA Geforce 9400M GPU and an Intel Core 2 Duo 2.2Ghz CPU, a lower-end card that is comparable to modern mobile GPUs. We tested our approach on various challenging 3D scenes, of which we analyzed three in detail. The SIBENIK scene is fully dynamic with quickly changing lighting, but it has relatively simple geometry and depth (plus motion) compresses well using our scheme. The SPONZA scene consists of mostly static diffuse geometry, for which spatial upsampling is strongly beneficial assuming that depth is encoded precisely enough. However, it represents a worst-case scenario for depth compression because of the detailed tree model. Finally, the FAIRY scene is excessively textured and exhibits high-tessellated dynamic objects and complex motion trajectories.

In Table 4.1, we analyze the performance of our method and unfold the timings for each step of our algorithm. A complementary performance results, where we show detailed timings of each component are presented in Table 4.3. As illustrated in Figure 4.2, most of these steps map to GPU pixel shaders, only the H.264 video and final edge encoding are CPU based. Edge image encoding performs neighborhood matching on the GPU, writing the result in a texture that is read back to main memory, where we apply CPU arithmetic coding. Despite this CPU involvement, we already achieve real-time performance on mainstream hardware for server and client. Our client-side timings are achieved in milliseconds. Even on a weak client (notebook with low-end GPU), our decoding and upsampling at resolution $800 \times 600$ reaches realtime frame rates of 25–30 fps (15.5 ms for diffusion and

| Scene | Bandwidth | | H.264 | | Our method | |
|---|---|---|---|---|---|---|
| | shading [*Mbit/sec*] | depth [*Mbit/sec*] | **A** [*dB*] | **B** [*dB*] | **A** [*dB*] | **B** [*dB*] |
| Sibenik | 2.0 | 1.0 | 35.1 | 49.6 | 35.7 | 49.5 |
| Sponza | 2.0 | 2.5 | 34.2 | 51.2 | 35.8 | 54.2 |
| Fairy | 2.0 | 2.0 | 30.6 | 52.5 | 33.5 | 64.2 |

**Table 4.2** – Quality comparison with H.264 codec

*For a given bandwidth constraint we compare the quality (PSNR measured on final output images) of our solution in respect to H.264 based depth+motion encoding.* ***A****) Spatio-temporal upsampling done on the client compared to ground truth high-res. image.* ***B****) Depth buffer reconstruction on the client side.*

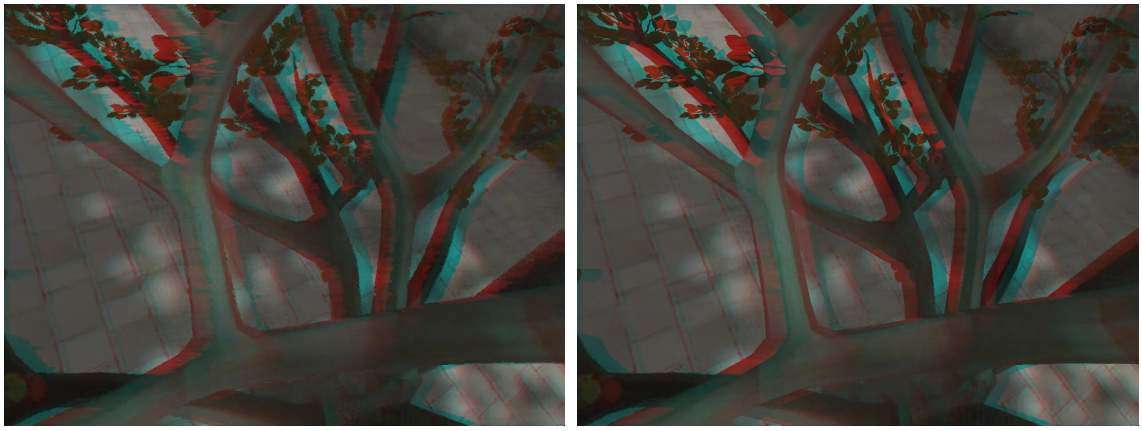

**Figure 4.7** – 3D stereo warping quality comparison

*Precise depth values along geometric discontinuities are crucial for correct and high-quality image warping. Left image shows warping result using frame and depth encoded with H.264 codec at 4 Mbit/sec. Imprecise edge representation generates visible distortions in reprojected image. Our solution (right image) addresses this issue by employing a custom depth/motion compression scheme that prioritizes values along the edges (3.8 Mbit/sec).*

upsampling on GPU, 25 ms for stream decoding on CPU). Moreover, the compression of motion and depth edges is relatively simple to perform for the server. Our scheme is suitable for low-cost client hardware and ready for today's portable technology. The server workload scales well for more clients by various means, particulary the upsampling factor enables strong speedups (see also supplementary material).

The video encoding of (low-res.) frames is performed on the highly optimized x264 CPU encoder. However, encoding performance depends on profile settings from which only "ultrafast" to "veryfast" settings achieved real-time rates on our hardware for HD-ready resolutions. Since we do not tolerate frame delay we can only use I- and P-frames (i.e., disable bidirectional motion-compensated B-frames) which, in combination with the two mentioned settings, yields poor quality for moderate bit rates (2 Mbit/sec). However, when encoding low-res. frames (as needed by our solution), "medium" or even "slow" quality settings achieve real-time performance.

| Scene | Resolution | Server | | | | | | | | | | Client | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Brute-force | | | Our Method | | | | | | | | | | | | |
| | | $T_{shade}$ | $T_{h264}$ | Fps | $U_{ratio}$ | $T_{geom}$ | $T^{low}_{shade}$ | $T_{prep}$ | $T_{edge}$ | $T_{enc}$ | $T^{low}_{h264}$ | Fps | Speedup | $T^{low}_{h264-1}$ | $T_{predict}$ | $T_{dec}$ | $T_{dif}$ | $T_{up}$ | Fps |
| Sibenik | SVGA | 57.2 | 32.3 | 10.7 | 2×2 | 4.1 | 16.0 | 2.6 | 1.8 | 10.3 | 12.5 | 21.1 | ×2.0 | 2.0 | 0.9 | 9.9 | 1.7 | 1.2 | 63.7 |
| | | | | | 4×4 | 4.1 | 4.5 | 2.2 | 1.6 | 17.5 | 9.1 | 25.6 | ×2.4 | 1.0 | 0.9 | 17.1 | 1.3 | 1.2 | 46.5 |
| | HD 720p | 106 | 46.0 | 6.3 | 2×2 | 5.9 | 29.0 | 3.7 | 3.6 | 16.4 | 18.3 | 13.0 | ×2.1 | 3.1 | 1.7 | 16.1 | 2.2 | 2.3 | 39.4 |
| | | | | | 4×4 | 6.0 | 8.5 | 3.2 | 3.0 | 17.1 | 9.7 | 21.1 | ×3.3 | 2.0 | 1.7 | 16.6 | 1.5 | 2.3 | 41.5 |
| | HD 1080p | 251 | 73.7 | 3.0 | 2×2 | 10 | 67.5 | 7.3 | 7.9 | 31.0 | 32.8 | 6.4 | ×2.1 | 7.2 | 3.9 | 30.5 | 3.4 | 4.8 | 20.1 |
| | | | | | 4×4 | 10.7 | 18.4 | 7.3 | 6.6 | 33.5 | 13.6 | 11.1 | ×3.7 | 2.2 | 3.7 | 32.5 | 3.4 | 4.8 | 21.5 |
| Sponza | SVGA | 63.0 | 32.4 | 10.0 | 2×2 | 4.6 | 16.7 | 2.6 | 1.9 | 20.8 | 12.7 | 16.9 | ×1.7 | 2.0 | 1.0 | 20.8 | 1.5 | 1.2 | 37.7 |
| | | | | | 4×4 | 4.6 | 4.5 | 2.3 | 1.6 | 20.5 | 9.2 | 23.4 | ×2.3 | 1.0 | 1.1 | 21.3 | 1.5 | 1.2 | 38.3 |
| | HD 720p | 120 | 45.0 | 5.8 | 2×2 | 6.6 | 31.9 | 4.0 | 3.6 | 21.5 | 19.2 | 11.5 | ×2.0 | 3.0 | 2.0 | 21.1 | 2.1 | 2.2 | 32.9 |
| | | | | | 4×4 | 6.6 | 8.5 | 3.9 | 3.0 | 25.5 | 9.7 | 17.5 | ×3.0 | 2.1 | 1.7 | 25.1 | 2.1 | 2.2 | 30.1 |
| | HD 1080p | 271 | 76.4 | 2.8 | 2×2 | 11.0 | 71.4 | 8.0 | 7.9 | 40.9 | 33.2 | 5.8 | ×2.1 | 7.1 | 4.1 | 39.7 | 3.9 | 4.8 | 16.8 |
| | | | | | 4×4 | 11.9 | 18.8 | 7.8 | 6.6 | 43.0 | 14.0 | 9.8 | ×3.5 | 2.1 | 3.8 | 42.1 | 3.9 | 4.7 | 17.7 |
| Fairy | SVGA | 31.6 | 31.0 | 14.9 | 2×2 | 4.4 | 8.7 | 2.4 | 1.9 | 19.0 | 12.1 | 20.6 | ×1.4 | 1.9 | 0.9 | 17.5 | 1.4 | 1.1 | 43.9 |
| | | | | | 4×4 | 4.4 | 2.8 | 2.2 | 1.6 | 15.0 | 9.2 | 28.4 | ×1.9 | 1.0 | 0.9 | 14.1 | 1.3 | 1.1 | 54.3 |
| | HD 720p | 51.2 | 44.4 | 9.9 | 2×2 | 5.8 | 15.2 | 4.3 | 3.6 | 17.3 | 18.4 | 15.5 | ×1.6 | 3.0 | 1.7 | 16.7 | 2.6 | 2.1 | 38.3 |
| | | | | | 4×4 | 5.9 | 4.7 | 4.3 | 3.0 | 22.7 | 9.6 | 19.9 | ×2.0 | 2.1 | 1.7 | 20.8 | 2.6 | 2.1 | 34.1 |
| | HD 1080p | 107 | 72.2 | 5.3 | 2×2 | 9.3 | 28.7 | 7.5 | 7.9 | 34.7 | 30.3 | 8.4 | ×1.6 | 7.5 | 3.9 | 32.1 | 3.6 | 4.7 | 19.3 |
| | | | | | 4×4 | 9.4 | 7.3 | 8.7 | 7.0 | 41.0 | 13.6 | 11.5 | ×2.2 | 2.0 | 5.9 | 38.8 | 2.8 | 4.7 | 18.5 |

**Table 4.3** – Detailed Server and Client timings

*Timings (in msec) for different target resolutions:* $800\times600$ *(SVGA),* $1280\times720$ *(HD 720p),* $1920\times1080$ *(HD 1080p) compared to reference (full-res.) rendering* $T_{shade}$ *and following video-encoding step* ($T_{h264}$) *using H.264 with disabled frame delay. The geometry-processing costs* ($T_{geom}$) *are the same for reference and our method. However, in our examples rendering costs are dominated by frame size due to complex pixel shading and hence, we give timings for different upsampling factors* ($U_{ratio}$). *In our system the server-side timings comprise: the geometry pass* ($T_{geom}$) *including the rendering preprocess (updating animation, shadow maps, etc.), the (deferred) low-res. pixel shading* ($T^{low}_{shade}$), *the preparation pass simulating the client steps* ($T_{prep}$), *which is initiated before the edge detection and encoding (i.e., previous depth/motion frame warping and edge diffusion), the edge selection* ($T_{edge}$) *which involves computing the perceptual edge significance in the low-res. shading image (up to this point all steps are performed on the GPU), our edge encoding including entropy coding on the CPU* ($T_{enc}$), *the H.264 low-res. frame encoding* ($T^{low}_{h264}$). *On the client side we perform: low-res. H.264 stream decoding* ($T^{low}_{h264}$), *previous depth/motion frame warping and edge prediction* ($T_{predict}$) *on GPU, edge image decoding* ($T_{dec}$), *edge diffusion* ($T_{diff}$) *followed by upsampling the low-res. video-frame* ($T_{up}$). *Note that several steps in the system pipeline are independent and could be executed in parallel.*

The resulting compression ratio and the quality is analyzed in Table 4.2. We compare the efficiency of our method for 3D stereo vision and spatio-temporal upsampling for various configurations. Regarding the absolute compression error (PSNR) of depth images, we are not always better than depth encoding with H.264. However, our compression scheme better preserves depth and motion vectors along edges, which is crucial for our applications. Hereby, we avoid visible distortions while in other homogeneous regions our introduced error is less perceptible. This property also translates in an improved frame reconstruction as illustrated in Figures 4.7 and 4.8, Table 4.2 and the accompanying video.

Generally, our method provides a high-quality depth and motion approximation well suited for upsampling and warping frames (see Figure 4.1). Our client-side spatio-temporal bilateral upsampling (Section 4.3.3) ensures robust upsampled frames even for a larger depth/motion-residual quantization (e.g., $q = 3$) since the edge topology is encoded loss-

**Figure 4.8** – Spatio-temporal upsampling quality comparison

*Here we show another type of visual distortion when using a DCT-based video codec for depth/motion compression. Decompressed motion vectors include small errors that cause temporal reprojection to warp pixels to bad locations. This effect propagates over time and results in noisy pixels and smearing in high contrast areas. As we store precise values at depth discontinuities our approach (right image) is not affected by the aforementioned artifact. Bandwidth settings are the same as in Figure 4.7.*

less and therefore avoids leakage between discontinuous image regions. In our results we encoded depth with 10 bits (or 8 bits when comparing against H.264) and 2D motion using 2 times 8 bits whereas for H.264-encoded depth we were limited to 8 bits only. H.264 offers a special *High* 4 : 4 : 4 *Predictive* mode particularly suited for depth, but featuring lossless encoding makes it impractical in our context.

Our compression exceeds the quality of the state-of-the-art platelets approach [MMS*09] (Figure 4.9). For a fair comparison, we only relied on an intra-frame context and did not make use of previous frames for this comparison. In particular, we manage to maintain complex boundaries, which are only linearly approximated for platelets. A disadvantage of our compression is that the bitrate is harder to control than for platelets. Nevertheless, our solution is practical and the adaptive scheme steers the bitrate convincingly to reach an acceptably low deviation (Figure 4.10). Such a property is important for online contexts.

Our diffusion solution compares favorably to many existing methods. Simple push-pull solutions fill holes using nearest samples, often involving a Gaussian pyramid. However, averaging across edges leads to leakage which might be unacceptable. For our solution, leakage occurs only where it is least perceivable (even for 3D warping) and has little impact on the upsampling process. PDE-based diffusion [FFLS08, OBW*08] provides good quality and smooth results, but is rather costly and difficult to control the number of iterations until convergence. Further, such a global diffusion is not needed and is disturbed by the uniform edge samples. Recent work by Fattal [Fat09] introduces second generation wavelets for edge-stopping smoothing and diffusion, that could be used in our context. Nonetheless, our approach is simpler and faster, with directly controlled execution time.

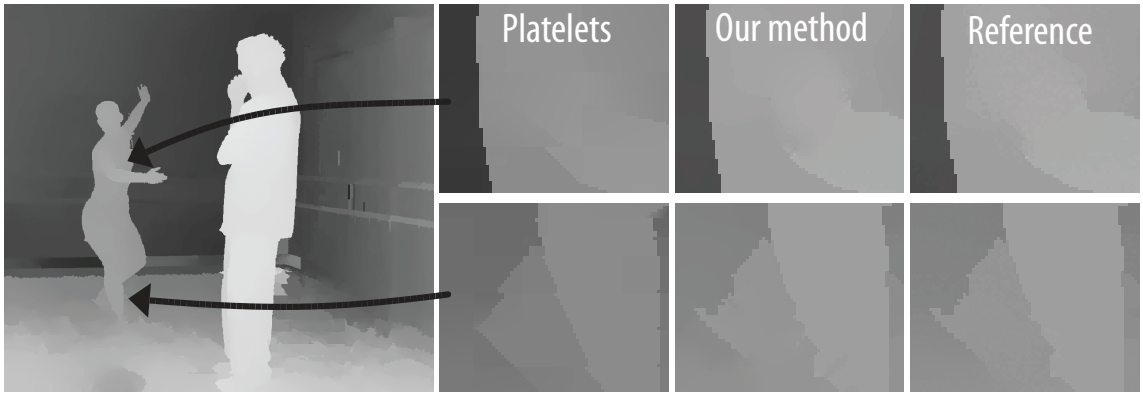


**Figure 4.9** – Our method vs platelets

*Depth reconstruction quality: For the same bandwidth (0.1 bits/pixel), our method achieves significantly better reconstruction quality (PSNR of 48.1 dB) than platelet encoding (PSNR of 43.9) and captures smooth regions and discontinuities. (Data courtesy of [MMS*09])*

Furthermore, our method achieves the highest-quality reconstruction when applied to our streamed representation, as it is inherently well suited for the additional sparse uniform sampling. Figure 4.4 compares the various approaches.

## 4.7 Limitations and Future Work

Our streaming solution inherits limitations from upsampling methods. For volumetric effects, mirror reflections, refractions or multi-sample rendering (e.g. antialiasing, transparency) the computed geometric flow will not correspond to the actual optical flow, thus, forcing temporal reprojection to fail. Deriving correct motion flow and depth in such a scenario is challenging. However, as shown in Figure 4.11, our scheme handles these situations partially by automatically falling back to pure spatial upsampling, more advanced detection methods or handling remain future work.

The H.264 video encoder uses the x264 library [x26], which is limited to 4:2:0 chroma subsampling that stores color data downsampled by a factor of 2 in both dimensions. This sometimes leads to undesired color bleeding in colorful scenes in case that the spatial upsampling window is larger than 2. However, we believe that 4:2:2 or even better 4:4:4 chroma subsampling would solve this issue.

Currently, we use a sparse regular sampling to account for low-frequency information. Although this proved sufficient for our purposes, an interesting alternative could be to rely on multi-scale edge detection. Further, our statistical edge encoding could be improved with a global optimization, but then we would lose efficiency.

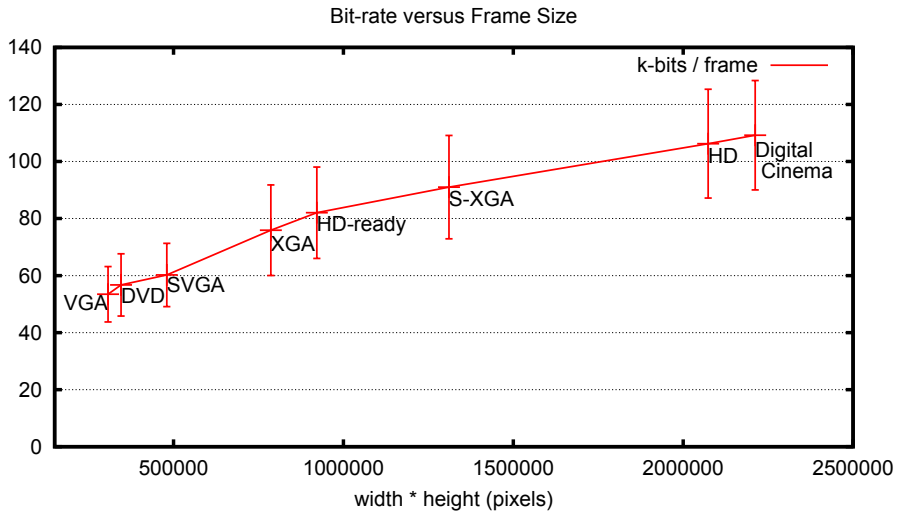


**Figure 4.10** – Bit rate vs frame resolution

*Plot of mean and standard dev. (over 400 frames) of the bit rate with respect to frame resolution for constant edge detection thresholds of the animated (camera, lighting and objects) Sibenik scene.*

Lossless edge-topology encoding is difficult to combine with constant bandwidth usage. For some particular scenarios the perceptual edge weighting scheme did not converge to the target rate quickly enough. Faster bandwidth control could be achieved by scaling the resolution and/or framerate dynamically, as in the H.264 SVC standard.

Performance-wise the proposed solution assumes server-side pixel computations to be moderately expensive. In case of simple shaders (Table 4.1, FAIRY scene) the system is not able to efficiently amortize the cost of additional computations with lower resolution rendering. However, similar to commercial solutions, video compression could be performed in hardware and further code optimizations (vectorization/parallelization) remain possible. In fact, our method could easily benefit from specialized hardware. Similar to H.264, the final (and the most expensive) encoding step employs a standard arithmetic encoder (i.e. CABAC) which makes us believe that a full-quality hardware realization is possible using existing components.

## 4.8 Conclusion

We presented an efficient solution for 3D-rendered-content streaming. Our method achieves similar compression as high-end encoders. In contrast to competition, our approach provides the client with additional information, such as depth, at no additional transmission cost. Our contribution is also the careful selection of such data: inexpensive to compute on the GPU server (much cheaper than full-resolution frames), but easy-to-compress with-

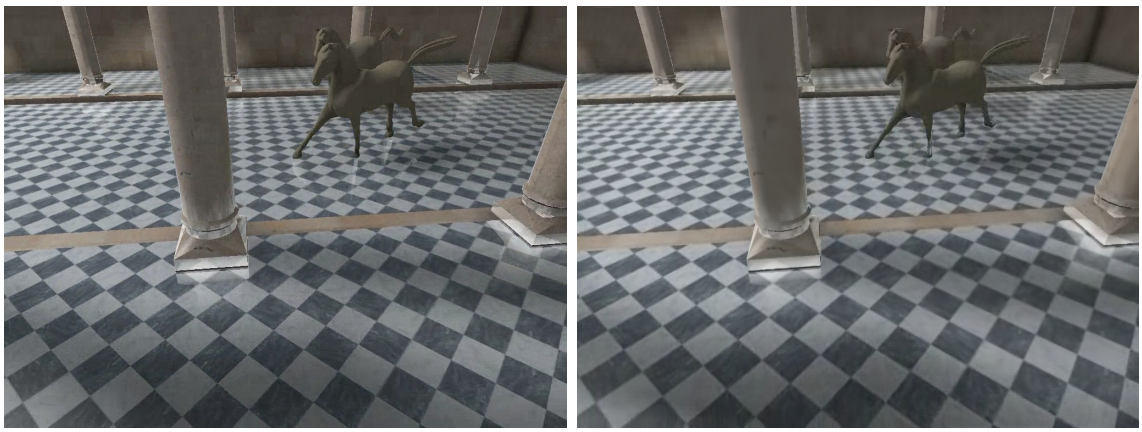

**Figure 4.11** – Final video quality comparison

*Here we show a scene featuring most difficult configuration for our architecture: dynamic lighting, dynamic geometry and mirror reflections on the floor and pillars. Our method (right image) still delivers acceptable quality with 2.8 Mbit/sec when comparing it to the reference encoded with H.264 (left image) with 3.0 Mbit/sec. Please note that our upsampled frame in particular the view-dependent reflections in the pillar appear more blurry. However, this is less noticeable when watching the accompanying video sequences.*

out dependence on future frames. Moreover, the full resolution depth maps enable 3D stereo viewing and the motion-flow maps can be used to boost the frame rate (important to reduce hold-type blur for LCD displays) or recover corrupted frames. Another interesting application is personalized rendering for advertisement or enhancement purposes. Our scheme is naturally compatible with existing game engines. It relies on specialized rendering procedures, but the required changes are minimal and easy to realize in game engines (for games, deferred shading is common and directly provides the needed data). This feature makes our approach attractive for many streaming contexts. Moreover, we are not limited to rasterization. Our framework can be beneficial for raytracing, where remote rendering is even more appropriate since computations require high-performance computers, or even cluster.

# Visual Maladaptation in Contrast Domain

In this work we simulate the effect of the human eye's maladaptation to visual perception over time through a supra-threshold contrast perception model that comprises adaptation mechanisms. Specifically, we attempt to visualize maladapted vision on a display device. Given the scene luminance, the model computes a measure of *perceived* multi-scale contrast by taking into account spatially and temporally varying contrast sensitivity in a maladapted state, which is then processed by the inverse model and mapped to a desired display's luminance assuming perfect adaptation. Our system simulates the effect of maladaptation locally, and models the shifting of peak spatial frequency sensitivity in maladapted vision in addition to the uniform decrease in contrast sensitivity among all frequencies. Through our GPU implementation we demonstrate the visibility loss of scene details due to maladaptation over time at an interactive speed.

## 5.1 Introduction

It is consciously experienced by everyone that intense changes in illumination temporally cause a loss in visual sensitivity that is later recovered over a time period. In fact, considering the highly variant and temporally changing real world illumination, the human visual system (HVS) is virtually never fully adapted in practice. Due to this *maladaptation*, the visibility of some scene regions is reduced which would otherwise be perfectly visible if the HVS was fully adapted.

The temporal loss of visibility can often be tolerated in daily life, since a large fraction of sensitivity is recovered relatively fast in just a few seconds through neural mechanisms, and most real world objects are purposely designed to be strongly visible. However, some tasks require quick reaction times and undiverted attention. For those the rate of adaptation may not be sufficient. For instance, a car driver entering a forest highway after driving against the sun can be temporarily blinded for a short amount of time jeopardizing safety. In fact, computational methods have been proposed to determine the magnitude of vehicle display visibility under dynamic lighting conditions [KSY92, Sil03] enabling the validation

of vehicle ergonomics and safety at design time. A more extreme case are fighter pilots who are exposed to much more drastic illumination changes, but regardless need to maintain near instant reaction capability at all times. On the other hand, the quickly recovered sensitivity may not be sufficient in environments containing low contrast objects. As an example, people often struggle to find their seats if they enter a movie theatre after the session started, while during the course of the movie the obstacles in the room become gradually visible due to the additional sensitivity recovery through the slower adaptation mechanisms based on chemical processes.

The aforementioned examples can benefit greatly from faithfully simulating the effect of maladaptation on visibility. Such a model should predict the visibility magnitude of both near– and supra-threshold scene details. Recent luminance based models [PFFG98, PTYG00, IFM05] tend to explicitly focus on modeling maladaptation while ignoring other HVS aspects such as contrast sensitivity and visual masking. The modeling of the latter mechanism [FPSG97] requires a contrast based approach involving a transducer function. Current contrast domain frameworks, however, often do not account for luminance adaptation and contrast sensitivity, as well as the overall sensitivity loss and shift in peak sensitivity due to maladaptation.

An important consequence of maladaptation is the locality of resulting visibility loss in a scene. For instance, looking outside the window in a dark room on a sunny day, one will eventually adapt to the bright illumination outdoors and start seeing objects clearly. If at that instance the gaze is directed towards the interior, the observer will not be able to discriminate objects that are visually less apparent. Thus, at any given time details in some scene regions are less visible than others, as dictated by the current level of maladaptation. The problem is that it is often not possible to predict the direction the gaze will be shifted towards, and thus the illumination levels that will be observed in the next timestep. Similar to real-world scenes, the emerging HDR displaying technology coupled with the ever increasing size of display devices is also prone to such local losses of visibility. From an application perspective it is beneficial to simulate how the entire scene would look like under current adaptation conditions, which is not possible using current methods relying on a single adaptation level for the entire scene.

We present a system that renders a series of images of a scene as it would be seen by a maladapted eye over time. Each separate image corresponds to the visual perception of the scene at a time step while the sensitivity is recovered. The time course of adaptation is modeled by considering both neural mechanisms and pigment bleaching and regeneration. Our framework operates in contrast multi-scale domain and models supra-threshold effects like visual masking, while also accounting for contrast sensitivity and luminance (mal)adaptation usually considered only in luminance domain frameworks. We also model the shifting of peak frequency sensitivity in maladapted vision, which has not

been considered by previous models. In the rest of the chapter we first discuss related work (Section 5.2), followed by a new model for simulation of human maladaptation in contrast domain (Section 5.3). Next, we present, analyze and discuss the results of our system (Section 5.4) and finally we conclude and suggest ideas for future research (Section 5.5).

## 5.2 Related Work

Previous models of time-course adaptation often operate on luminance and are not able to simulate visual phenomena locally. In this work, our goal is to simulate local maladaptation in *contrast domain* to account for supra-threshold mechanisms of vision as well as near-threshold. There were a few elaborate models of contrast perception proposed in history, but a vast majority of those were not concerned with the simulation of the time-course of maladaptation.

Ferwerda et al. [FPSG96] presented a computational model of visual adaptation. Their model captures the changes in threshold visibility, color appearance, visual acuity, and sensitivity over time using Ward's scaling tone mapping approach [War94]. Ward's mapping is enriched by an offset parameter that is a function of time. The visual acuity is approximated by removing higher frequencies according to Schaler's measurements. This is a simplistic approach because human sensitivity to contrast also decreases for lower frequencies. A photoreceptor-based global time-dependent tone mapping method presented by Pattanaik et al. [PTYG00] is built on parts of an advanced Hunt's model of color vision [Hun95]. By means of the adaptation model the method accounts for time dependency of retinal adaptation mechanisms for both cones and rods. However, as this adaptation model and the method itself are global they can simulate neither local adaptation mechanisms nor human contrast sensitivity. Irawan et al. [IFM05] devised a model of low vision that is able to simulate the performance of an impaired or aged human visual system. The model is based on the combination of histogram adjustment [LRP97] and Pattanaik et al.'s [PTYG00] global tone mapping methods. Due to the maladapted threshold-versus-intensity function (*tvia*), it can mimic the viewer's changing adaptation. The method is able to simulate the effect of maladaptation, but only at the threshold level and only globally for the whole image. However we are also interested in the supra-threshold effects of maladaptation in addition to visual perception around the threshold level.

Pattanaik et al. [PFFG98] proposed an advanced multiscale model of adaptation and spatial vision. As the model is based on spatial decomposition it can predict spatial contrast sensitivity behavior. The authors proposed gain functions that should be valid both for near– and supra-threshold luminance levels. However, the model does not comprise the time course of adaptation and is therefore unable to simulate effects of maladaptation.

More recently, Mantiuk et al. [MMS06] proposed a multiscale framework for perceptual processing of contrast. The method simulates supra-threshold perception (compression) of contrasts on multiple scales using transducer functions. However, contrasts still need to be compressed in a response space and yet another and more artificial compression is accomplished by the optimizer due to its weighting coefficients. The output of the optimization problem solver is therefore hard to calibrate for the correct reproduction of luminance perception. Moreover, the method in fact does not simulate human adaptation.

HVS models involving maladaptation have been also been proposed in the context of detecting the visibility loss on display devices due to dynamically changing illumination [AMS09]. While in this work temporal maladaptation is modeled in the contrast domain, they consider global adaptation and only output a "visibility map" that depicts distortions in the image structure similar to image quality assessment metrics, instead of rendering images of the scene appearance in a maladapted state. Furthermore, they don't model the change in spatial frequency sensitivity due to maladaptation, which we discuss in detail in Section 5.3.2.

### 5.2.1 Human Contrast Sensitivity in Maladapted State

Vision literature concerning the modeling human spatial *contrast sensitivity in an adapted state* usually through a *contrast sensitivity function* (CSF) is rich [Bar99]. Much work has also been done on *temporal contrast sensitivity* [Kel74], i.e. the sensitivity of HVS to the spatial frequencies over time, as this (and so called critical flicker frequency) was crucial in the design of first CRT display devices. However, measurements of CSF in maladapted states are hardly that obvious, perhaps due to the complicated testing and evaluation process. Maladapted luminance intensity thresholds are measured only for simple stimuli without any variation of spatial frequency [Dow87]. Consequently, in the rest of this section we discuss findings on the shape of CSF in maladapted conditions.

The encoding of contrast within the human visual system is thought to be mediated by two processing streams: the magnocellular (M) and parvocellular (P) pathways [Len93]. To investigate the effect of the pathways, Lenova et al. [LPS03] and Alexander et al. [ABF*04] measured contrast sensitivity using two different paradigms. In the steady-pedestal paradigm, they briefly presented a test stimulus against a continuously presented adaptation field. In the pulsed-pedestal paradigm, the test stimulus was presented simultaneously with the adapting field. The steady-pedestal paradigm favors the M pathway, while the pulsed-pedestal paradigm favors the P pathway. The measured mean contrast sensitivity function for control subjects for steady-pedestal has a low-pass shape, while for the pulsed-pedestal it has a band-pass shape, see Figure 5.1 (left).

On the other hand, Hahn et al. [HG95] found the CSF to be invariant in shape during dark

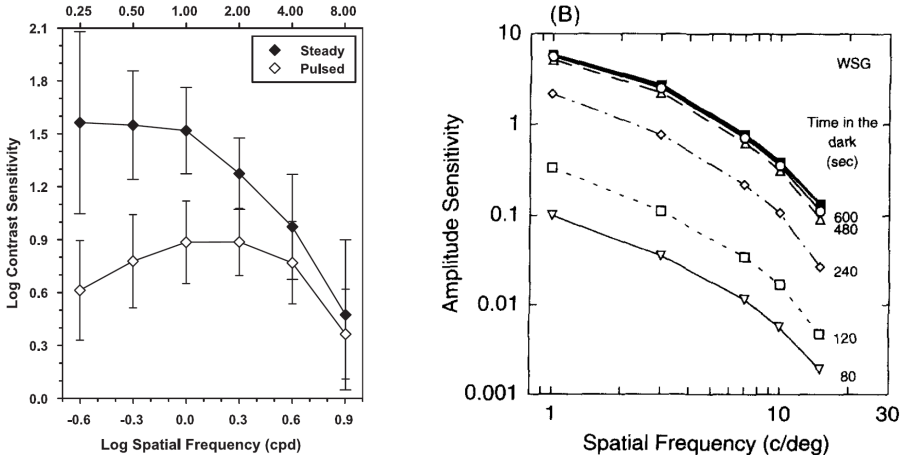


**Figure 5.1** – Measurements of maladapted contrast sensitivity

*Left: the shape of CSF for steady (adapted) state and for briefly pulsed (maladapted) stimulus, adapted from [ABF*04], right: amplitude sensitivity functions during dark adaptation, adapted from [HG95] (the right image shows the amplitude sensitivity functions (ASF); one can obtain CSF from ASF by multiplying with the background luminance).*

adaptation. Differently from Leonova et al. [LPS03] who presented stimuli only briefly to observers during experimentation, Hahn et al. measured a longer time course of dark adaptation ranging from seconds to hundreds of seconds, see Figure 5.1 (right). This suggests that the transition from original to destination stimuli is very fast in terms of sensitivity to spatial frequencies (as modulated by P pathway), but much slower in terms of overall sensitivity to contrast. In other words, the shift in frequency sensitivity happens almost instantly and is retained during the time course of adaptation to the destination stimulus.

In our method, we use Daly's CSF [Dal93, MDMS05] and we were tempted to simulate the aforementioned transition behavior by using current adaptation luminance as input parameter $L_a$ of the maladapted observer. This approach, combined with the use of *maladaptation ratio* resulted in reasonable time course shape of the CSF. (The maladaptation ratio is approximated as $cvi(L_b)/cvia(L_b,L_a)$, where *cvi* and *cvia* are the contrast versus intensity functions for adapted and maladapted eye, respectively, $L_a$ is the current adaptation luminance, and $L_b$ is the current background luminance [AMS09]). However, this method implies the assumption that the spatial frequency sensitivity characteristics of the HVS remains constant in maladapted states, since the CSF we use was measured for the adapted eye. We can neither calibrate nor justify this approach as we were not able to find a sufficient amount of maladapted CSF experimental measurement data.

Therefore, in our model we incorporate the shift in frequency sensitivity due to maladaptation to the maladaptation ratio approach. Following an abrupt illumination change, we instantly modify the shape of the CSF to reflect the spatial frequency sensitivity in the

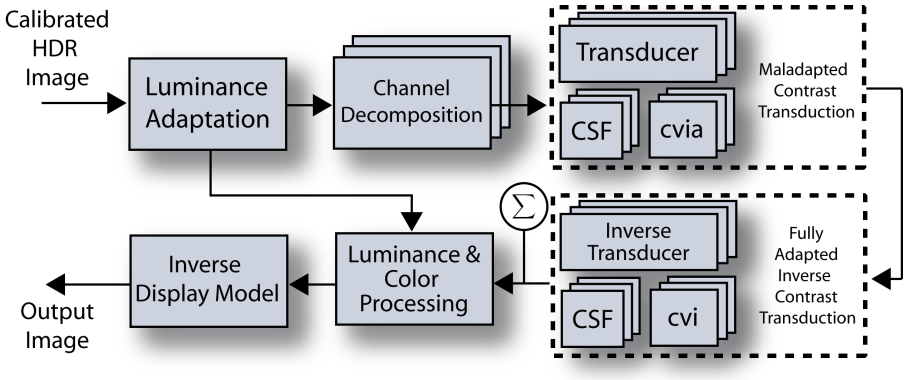


**Figure 5.2** – Flow chart of the proposed method
*See text for details.*

target state, and then increase the sensitivities globally using the maladaptation ratio over the time course of adaptation. Our method is supported by experimental evidence: the sensitivity after sudden illumination change drops down drastically and when it is (at least partially) regenerated the curve already has the invariant shape of the target (compare Figure 5.1 right with Figure 5.5 right).

## 5.3 Simulation of Visual Maladaptation

The data flow of the proposed model for human contrast perception in maladapted states is illustrated in Figure 5.2 for the steady-state. We assume that the input HDR image is calibrated in $\mathrm{cd/m^2}$ units. First, we construct background luminance and local adaptation maps ($L_b$,$L_a$), which are used both for contrast processing and final display purposes. The adaptation map is modified over time to model the temporal adaptation. Simultaneously, we decompose the input image into the contrast representation ($C$) using the Laplacian pyramid [BA83]. We then process physical contrast by a model of *maladapted scene observer* depending on sensitivity to spatial frequencies as well as on the current adaptation state to get the perceptual contrast responses ($R$). The response values are transformed by inverse *adapted display observer* model to obtain physical display contrast. All contrast processing steps are performed on multiple scales simultaneously. Consecutively, the physical contrast is converted to display luminance map ($L_d$) and colors are processed ($I_{\mathrm{corr}}$). Finally, the inverse display model produces the output code values ($I_{\mathrm{out}}$) that are shown on the display device.

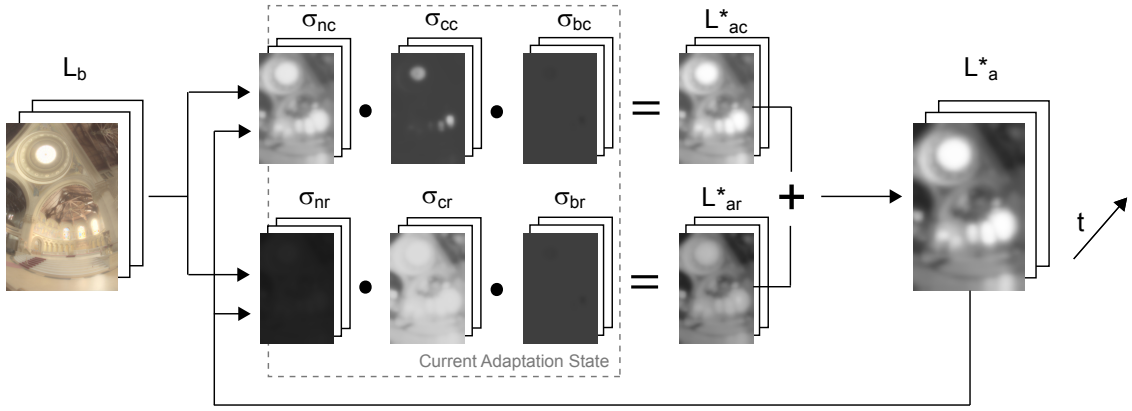


**Figure 5.3** – Visual illustration of the local luminance adaptation processing

*Temporal behavior is modeled for rod $L^*_{ar}$ and cone $L^*_{ac}$ adaptations separately, which are computed from adaptation maps ($\sigma_{bc}, \sigma_{cc}, \sigma_{nc}, \sigma_{br}, \sigma_{cr}, \sigma_{nr}$) that model various adaptation mechanisms. These adaptation maps are updated at each timestep using the current background luminance map $L_b$. In maladaptation computation (see Sec. 5.3.2), a compound adaptation map $L^*_a$ is obtained by adding adaptation maps for rods and cones.* (HDR image courtesy of Paul Debevec)

### 5.3.1 Adaptation Map

The adaptation map $L_a$ represents the actual state of local adaptation of the observer. The construction of the *local adaptation* map is based on the actual *background luminance map* $L_b$ and on the previous course of local adaptation (see Section 5.3.4 for the details on temporal adaptation). Background luminance $L_b$ is the actual stimulus of an observer and is calculated for each input frame as the blurred image of input luminance (as the contrast sensitivity function was measured for foveated vision we blur the luminance conformably to one visual degree (1°) [LRP97]). To accomplish this we use the Gaussian filter with the kernel size $K = \frac{2d}{p}\tan(\frac{\pi}{360})$, where $p$ is the pixel size (in meters) and $d$ is the observer's distance from the display (in meters).

Similarly to Irawan et al. [IFM05] we model the adaptation due to human rods and cones separately. To obtain a single response value, Hunt [Hun95, Sec. 31.8.2] proposed to sum the achromatic cone and rod responses up. The current adaptation luminance $L^*_a$ is therefore obtained (as also illustrated in Figure 5.3) as a sum of cone ($L^*_{ac}$) and rod ($L^*_{ar}$) adaptations: $L^*_a = L^*_{ac} + L^*_{ar}$, where $L^*_{ac} = \sigma_{bc} \cdot \sigma_{cc} \cdot \sigma_{nc}$ and $L^*_{ar} = \sigma_{br} \cdot \sigma_{cr} \cdot \sigma_{nr}$. Factor $\sigma_b$ accounts for the *photopigment bleaching and regeneration*: $\sigma_b(L_b) = 1/p(L_b)$, where $p(L) = I_0/(I_0 + L)$ and $I_0 = 10^4$ cd/m$^2$. To model *neural adaptation mechanisms*, we calculate $\sigma_n$ (fast neural adaptation) and $\sigma_c$ (slow neural adaptation) for rods and cones using the equations proposed by Irawan et al [IFM05]. Note however that our implementation of human adaptation is local (i.e., we have the adaptation map) and all of the factors mentioned above ($L^*_{ac}, \sigma_{bc}, \sigma_{cc}, \sigma_{nc}, L^*_{ar}, \sigma_{br}, \sigma_{cr}, \sigma_{nr}$) are not single values, but complete maps spanning the whole image.

**Figure 5.4** – Comparison of the effect of global and local adaptation

*Left: global adaptation (using global values $L_{ag}$ and $L_{bg}$), right: local adaptation (using local $L_b$ and global $L_{ag}$). Notice that local background luminance map allows to simulate different sensitivity to spatial contrasts according to varying illumination in the scene.*

For the subsequent processing (CSF filtering, *cvi*, *cvia* functions), we need to convert the adaptation values $L^*_{\cdot}$ scaled in hypothetical *perceptual adaptation* units back into the physical units. In other words, we are searching for an adaptation map $L_a$ in physical luminance units that would evoke the actual maladapted state $L^*_a$ in the observer's visual system. To do this, we numerically invert the function $L^*_a$ and set $L_a = L^{*-1}_a(L^*_a(L^*_{ac}, L^*_{ar}))$. Note that for the fully adapted observer this results in $L_a = L_b$ as expected, but for the maladapted observer, the behavior of this function is more complex (see Section 5.3.4).

For experimental visual analysis and illustration purposes we allow the use of the *global adaptation* value $L_{ag}$ instead of the local adaptation map. We can calculate the global background luminance $L_{bg}$ as a geometric mean of the input luminance $L$ for each pixel: $L_{bg} = (\Pi^n L)^{1/n}$ and similarly we obtain the global adaptation luminance $L_{ag}$. Global $L_{ag}$ is useful for the analysis of static images, where it would be hard to change local adaptation map $L_a$ manually if a refence HDR image depicting the adaptation state is not present. Note that in the rest of the figures the background luminance ($L_b$) is still local even in global adaptation ($L_{ag}$) case (see Figure 5.4-right), with the exception of Figure 5.4-left where we illustrate global adaptation $L_{ag}$ with global background $L_{bg}$ luminance for comparison.

### 5.3.2 Maladapted Spatial Sensitivity to Contrast

To account for sensitivity to spatial frequencies, we utilize the contrast sensitivity function (CSF) proposed by Daly [Dal93, MDMS05]. The corresponding spatial frequency $\rho$ (in c/deg) for each level $l$ (starting from 1) of Laplacian pyramid is obtained as $\rho = K/2^{(l-1)}$. The size of the image (in $X \times Y$ pixels) in visual degrees is $i^2 = \max(X,Y)/K$. Given spatial

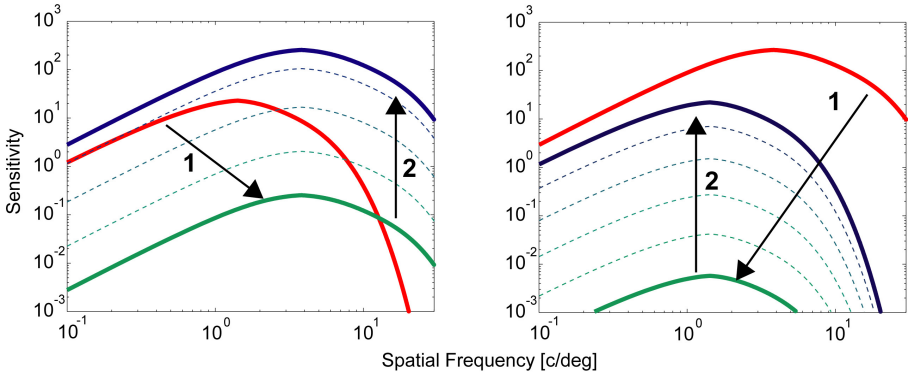


**Figure 5.5** – Simulation of time-course of contrast sensitivity for a maladapted observer

*Left: transition from dark to bright, Right: transition from bright to dark environments (dark adaptation).*

frequency $\rho$ in *cycles/deg*, observer distance $d$ in meters, image size $i^2$ in visual degrees, and current background luminance level $L_b$ (in $\mathrm{cd/m^2}$), and neglecting orientation $\theta$ and eccentricity $c$ we can calculate the sensitivity $S_a$ for contrast magnitudes $C$ (in Weber's units) for each pixel at each level $l$ of pyramid:

$$S_{al} = CSF(\rho, L_{bl}, \theta, i^2, d, c), \tag{5.1}$$

where the coarser background luminance map $L_{b(l+1)}$ is a downsampled from the finer scale map $L_{bl}$. We account for maladaptation by computing the maladaptation fraction as given in [AMS09]:

$$S_{ml} = S_{al} \cdot \frac{cvi(L_{bl})}{cvia(L_{bl}, L_{al})}, \tag{5.2}$$

where $S_{ml}$ is the sensitivity in the maladapted state, $S_{al}$ is the sensitivity at the fully adapted state, $L_{bl}$ is the current background luminance and $L_{al}$ is the current adaptation luminance. The subscript $l$ indicates the scale of each map.

Figure 5.5 shows the change in shape of CSF between two adapted states. In the left image, a subject is adapted to a dark environment. Accordingly, her sensitivity to contrast is low and shifted to low spatial frequencies (blue curve). After the exposition to a bright environment, the sensitivity rapidly shifts towards higher frequencies (arrow 1), but due to the maladaptation (as one is blinded by strong light for some time) the sensitivity is still very low (green curve). However, sensitivity is restored over time (arrow 2) to reach the final fully adapted state for the bright environment (red curve). The process is similar for a subject adapted to the bright environment (red curve in Figure 5.5 right). First, the sensitivity drops rapidly (arrow 1), shifts to the low frequencies (green curve) and consecutively it regenerates (arrow 2) to the final dark-adapted state (blue curve). The described behavior is in accord with psychophysical experiments conducted by Hahn and Geisler [HG95], who measured that the CSFs are nearly identical throughout the course

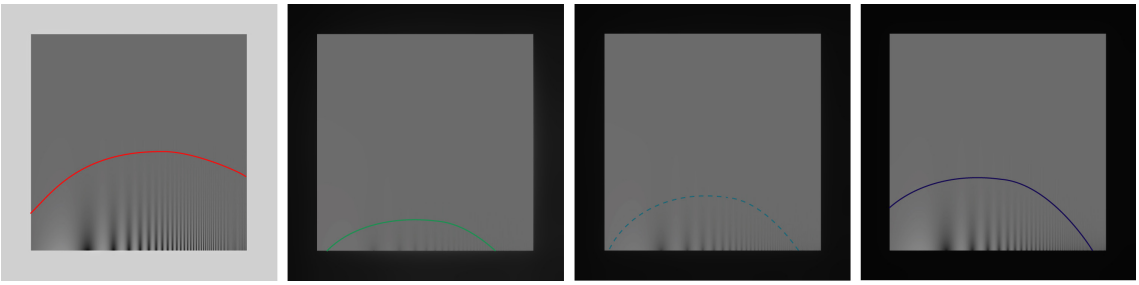

**Figure 5.6** – Classical Campbell-Robson contrast sensitivity chart for dark adaptation

*From left to right: (1) fully adapted state in a relatively bright environment (adaptation luminance 112 $cd/m^2$), (2) background luminance was decreased to 2 $cd/m^2$, the contrast sensitivity moves to lower frequencies, but due to maladaptation, it is basically very low, (3) sensitivity regenerates according to dark adaptation time-course, (4) final fully adapted state (adaptation luminance 3 $cd/m^2$). The curves show the thresholds observed from approximately 30 centimeters at original paper size.*

of dark adaptation. Naturally, the two processes differ in the speed of the sensitivity regeneration and we describe our implementation of the temporal aspects of adaptation below.

### 5.3.3 Contrast Transduction

The visual sensitivity to a contrast patch of a certain spatial frequency decreases with the presence of other similar frequency contrast. Daly's Visible Differences Predictor (VDP) [Dal93] accounts for this effect known as *visual masking* using a threshold elevation map. This approach trades off supra-threshold contrast interval for near-threshold precision. Such a trade-off is not suitable to our purposes, as real-world scenes are expected to comprise contrast well above the visibility threshold. Thus, in our model we employ the *transducer* function $T$ described in [Wil80] based on the premise that it is tuned for both near–and supra threshold precision. The contrast $C$ at each scale is processed separately as follows:

$$R_{al|ml} = T(C, S_{al|ml}) = \frac{3.291 \cdot [(1 + (S_{al|ml}C)^3)^{1/3} - 1]}{0.2599 \cdot (3.433 + S_{al|ml}C)^{0.8}}, \tag{5.3}$$

where $R_{al|ml}$ is adapted or maladapted human perceptual response to contrast, $S_{al|ml}$ is the sensitivity at either maladapted or fully adapted state. The constants are taken from Wilson's work without any change. The monotonically increasing behaviour of the transducer function enables a fast inversion through the use of a lookup table stored in GPU memory.

### 5.3.4 Temporal Adaptation

Temporal adaptation can be modeled through two separate exponential decay functions; one for *pigment bleaching and regeneration* and another for *neural adaptation* [IFM05]. For simplicity, we describe the adaptation process generically, but recall (Sec. 5.3.1) that the final adaptation map $L_a^*$ is combined from six values that possess different time constants.

The time course of the neural adaptation mechanism from perceived luminance $L_0^*$ at time $t = 0$, to $L_\cdot^*$ (where $L_\cdot^*$ is $\sigma_{cc}, \sigma_{nc}, \sigma_{cr}, \sigma_{nr}$) is modeled as follows:

$$L_\cdot^* = L_b^* + (L_0^* - L_b^*)\, e^{\frac{-t}{t_0}}. \tag{5.4}$$

The contribution of neural adaptation to temporal recovery of visual sensitivity is modeled by updating the *cvia* function at each time step using the current $L_a^*$. We set $t_0$ to 0.08 seconds for cones, and 0.15 seconds for the rods[IFM05].

Pigment bleaching and regeneration (modeled by $\sigma_{bc}$ and $\sigma_{br}$), unlike neural adaptation, are slow and not symmetric for dark and bright adaptation. Assuming that the amount of signal transmitted by receptors is proportional to $p \cdot L$, the fraction of unbleached pigments $p$ is computed as in Equation 5.5:

$$p = p(L_b) + (p_0 - p(L_b))\, e^{\frac{-t}{t_0 \cdot p(L_b)}}. \tag{5.5}$$

In the steady state, $p(L)$ is $I_0/(I_0 + L)$ where $I_0$ is $10^4$ $cd/m^2$. The time constant $t_0$ is set to 110 and 400 seconds for cones and rods, respectively.

### 5.3.5 Luminance and Color Processing

The inverse transducer converts maladapted contrast responses $R_m$ to the luminance values $L_m$. By summing all the levels of the Laplacian pyramid we obtain the maladapted luminance map. This map represents the hypothetical output of the display device that would evoke the same perception of *contrast* in a fully-adapted display observer as the original HDR scene in the maladapted observer. However, to account also for the luminance sensitivity, we transform $L_m$ using the S-shaped function as follows:

$$L_d = \frac{L_m}{L_m + \bar{L_a^*}}; \qquad \bar{L_a^*} = (\Pi^n L_a^*)^{1/n}, \tag{5.6}$$

where $L_d$ is the output display luminance, $L_m$ is the luminance value we obtained from of inverse observer model, and $\bar{L_a^*}$ is a geometric average of the current adaptation luminance map. Note that the value of $L_a^*$ accounts for the current (mal)adaptation state and therefore the S-shaped function results in dark images for dark adaptation scenario and bright images

for adaptation to bright scenes, and the sensation will improve according to the temporal adaptation as described above.

As our aim is the simulation of maladaptation in contrast domain (and not the simulation of color vision phenomena), we perform only very simplified color processing. Simply put, all the above described processing happens on achromatic channel only. However, as the local adaptation map $L_a$ is calculated as a combination of the human rod and cone responses ($L_{ar}, L_{ac}$, refer to Section 5.3.4) we can utilize them for color processing. In the absence of a model of color perception specialized in maladapted HVS states, we desaturate the colors as follows: $I_{\text{corr}} = I^s$ (for each color channel $I$ separately) using the following saturation coefficient $s = L_{ac}/(L_{ac} + L_{ar})$, where $L_{ac}$ and $L_{ar}$ are current cone and rod adaptation map values, respectively.

### 5.3.6 Inverse Display Model

Our simple display model consists of three parameters, the maximum and minimum display luminance, and a gamma value which we set to 1/2.2. The gamma corrected values $I_{\text{corr}}$ are fitted to the luminance range of the display by a simple linear mapping. Our interface allows the user to control the display luminance range, this way a variety of display types can be approximated. For a more precise simulation of specific displays the linear mapping can be replaced by the display response function.

## 5.4 Results

In this section we discuss our results, implementation details, and the other possible uses of the model. Please refer to supplemental materials (`http://mpi-inf.mpg.de/~mcadik/maladaptation`) for further results.

A visual verification of maladapted contrast sensitivity behavior (described in Sec. 5.3.2) is presented in Figure 5.6. We augmented the Campbell-Robson chart with a frame of uniform luminance and generated two different HDR images (an initial and a final) to simulate the dark adaptation. The results show both the shift in peak frequency and the drop and regeneration of absolute sensitivity as expected. As we want to illustrate only the contrast processing in this figure, we simplify the equation (5.6) to $L_d = L_m/(L_m + k)$, where $k$ is set to 100 $cd/m^2$. Otherwise the maladapted images (the two middle images in Figure 5.6) would be too dark to visualize. Compare the output of our model in Figure 5.6 with Figures 5.5 and 5.1 (right).

In Figure 5.7 we compare our results to the approach of Irawan et al. [IFM05]. The refer-

**Figure 5.7** – Comparison of our method to Irawan et al.

*We are simulating the fast adaptation from a dark environment* ($10^{-4}$ cd/m$^2$) *to the stained glass* (17 cd/m$^2$). *Top row: method of Irawan et al. can not simulate differences in perception of contrasts in bright and dark parts of the scene, however our method can (bottom row). Columns from left to right:* $t = 0.01$s, $t = 0.02$s, $t = 0.05$s, $t = 0.1$s,$t = 60$s *(fully adapted state).* (HDR image courtesy of OpenEXR)

ence method (upper row) is based on global tone mapping function and global background luminance ($L_{bg}$), while our approach (bottom row) operates on contrasts and utilizes local background luminance ($L_b$). In both cases global adaptation luminance $L_{ag}$ is assumed. Therefore our method accounts for diverse perception of bright and dark areas of the scene. One can notice a difference in the fully adapted state as well (Figure 5.7 – rightmost images): in our model, the stained glass window is reproduced sharply and all the details are visible, while the dark area below the desk is blurred, which is the expected behavior. Note that by considering local background luminance ($L_b$) we ignore changes in the state of adaptation due to attending different image regions as a consequence of the saccadic eye motion. We rather visualize the image appearance under the condition that the eye attends locally each respective region without any gaze change. Thus, Figure 5.7 (bottom row) presents a synthetic summary how each specific region will be seen under this assumption, but the overall image appearance may not be presented precisely. Irawan used another extreme approach by considering global background $L_{bg}$ (Figure 5.7 – upper row), in which case it is implicitly assumed that through the gaze direction changes the eye adaptation tends to some average luminance in the scene. Since the most dramatic changes in light adaptation take place during the time required just for a couple of fixa-

**Figure 5.8** – Rendering of the interior of an airport control tower

*Left: fully adapted state (*178 cd/m$^2$*). Right: maladaptation due to a previous exposition to the bright sky (*$10^4$ cd/m$^2$*),* $t = 0.5$s*. Compare the visibility of displays and controls in close-ups (bottom pair).* (HDR image courtesy of Greg Ward)

tions this assumption is also not realistic in particular for video, while it is commonly used. Thus, Figure 5.7 (upper row) gives perhaps a better prediction of overall image impression (except that no frequency processing is accomplished), but the local detail visibility might be better predicted in Figure 5.7 (bottom row).

In Figure 5.8 we illustrate an application of our approach to analysis of the visibility of a display and controls on the panel in a flight control room. Left column shows fully adapted state, where all the details are well visible. After the adaptation to the bright sky however, those details are not noticeable for some seconds.

Our system enables also to simulate even more complex scenario (see Figure 5.9) where in lack of opposing evidence we consider local adaptation $L_a$ and background $L_b$ maps, in which case each local region in the image has also corresponding local adaptation. Imagine an observer who is adapted to relatively bright illumination of a living room and then she instantly moves to her desk in a dim work-room. For a moment, while her sight is being regenerated, she does not see the details in some parts of the scene due to the previous adaptation to much brighter environment. The vision reaches the fully adapted state in some seconds, but due to the low illumination of the work-room, the vision is still not

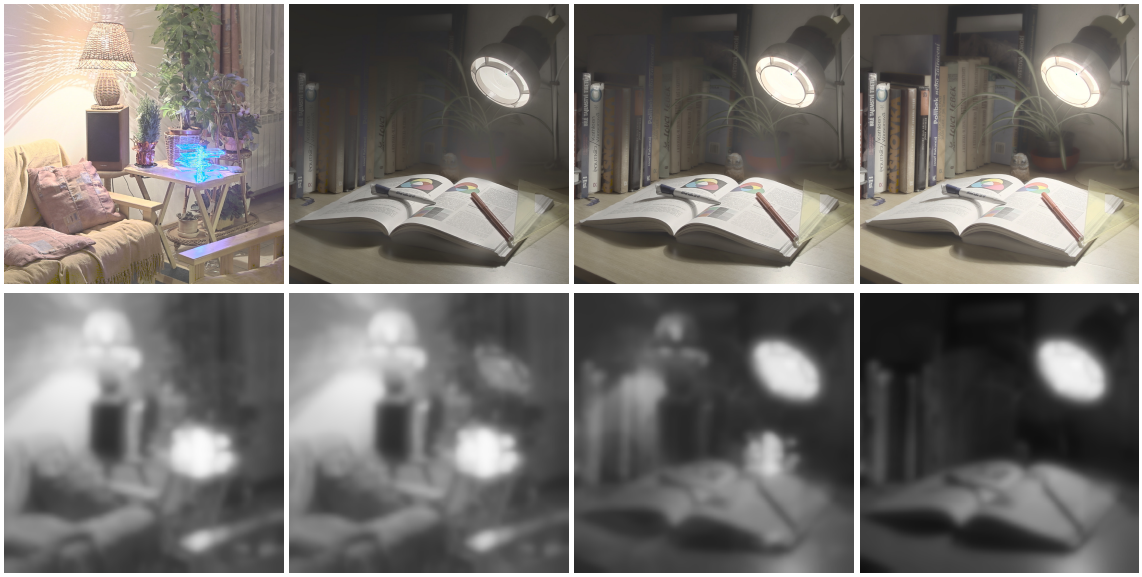

**Figure 5.9** – Simulation of maladaptation in a complex hypothetical scenario

*Top row, from left to right: (1) fully adapted state in a relatively bright living room (adaptation luminance 70 $cd/m^2$), (2) rapid movement to a work-room (8 $cd/m^2$), time $t = 0.02$s, (3) sensitivity regenerates and aftereffects diminish ($t = 0.1$s), (4) in $t = 0.5$s the observer reached nearly fully-adapted state, but some dark details are washed out, due to the dim illumination of the work-room. Bottom row: states of adaptation maps $L_a$ corresponding to the upper images. As the values in adaptation maps are HDR, they were tone mapped using the global version of Reinhard's [RSSF02] TMO for the display purpose.*

sharp in dark parts of the scene.

### 5.4.1 Fast GPU Implementation

In order to achieve real-time performance, we moved perceptual (contrast transduction, calculation of maladaptation, local adaptation of rods and cones) and image processing (laplacian pyramid, tone-mapping) parts of the algorithm to the GPU. Additionally, to enable real-time HDR movie processing we use the GPU for the decompression of frames stored in radiance format. The resulting application achieves interactive frame rates on a mainstream hardware: Intel Core2 3.0Ghz CPU, 4GB of RAM and NVidia GTX260 GPU. For 1024x1024 HDR image, our system generates a solid 75 FPS. Currently, the performance is limited by Java GUI components, that require a time-consuming texture transfer to the system memory. Eliminating this call, and displaying the texture directly could speed up the rendering even further.

**Figure 5.10** – Simulation of maladaptation in two LDR images

*In each pair: left: original LDR image. Right: maladaptation simulation using the background luminance from the HDR image (*200cd/m$^2$*) obtained by the inverse tone mapping. Simulated adaptation luminance:* 20cd/m$^2$. (HDR image courtesy of Allan Rempel et al. [RTS*07])

### 5.4.2 Simulating Maladaptation in LDR Images

Another possible application of our model is the simulation of maladaptation effects on contrast perception in an ordinary (LDR) image, see Figure 5.10. Let us assume that only an LDR image is available, but we want to know how its appearance will be affected due to the maladaptation. It is possible to perform inverse tone mapping [RTS*07] and derive a reasonable approximation of adaptation map. Having a scene referred HDR image as the adaptation pattern, we can simulate appropriate HVS reaction for arbitrary LDR image as follows: we run the model for the HDR image and we keep the contrast responses $R_a$ for fully adapted observer and for a particular maladapted state $R_m$, then we linearize the LDR image using inverse gamma correction and decompose it using Laplacian pyramid. To simulate maladaptation in the LDR image, we multiply the values in the Laplacian pyramid as follows:

$$C'_{i,j,l} = C_{i,j,l} \cdot \frac{R^m_{i,j,l}}{R^a_{i,j,l}}, \tag{5.7}$$

where $C$ is the current LDR contrast value for pixel $i, j$ and level $l$ of Laplacian pyramid. The final LDR image with the simulated maladaptation effect is obtained by adding $C'$ at all the levels of modified Laplacian pyramid. In this special case we simulate only the effect of maladaptation to the perception of contrasts, as we omit the luminance processing (i.e. we do not involve equation (5.6)).

## 5.5 Conclusion

We presented an efficient, real-time visual maladaptation framework capable of rendering images of a scene as perceived by a maladapted observer. Our model operates on contrast domain and accounts for supra-threshold HVS mechanisms such as visual masking, as well

as luminance adaptation and contrast sensitivity as a function of spatial frequencies that have often been neglected by previous contrast domain methods. We also model the shift in spatial frequency sensitivity due to maladaptation, which we found to have a significant effect on scene visibility. We discuss a fast GPU implementation that enables interactive rendering of maladapted images. Our system can potentially be used to simulate human vision in illumination conditions causing extreme maladaptation in real-world scenarios such as driving.

### 5.5.1 Limitations and Future Work

As the model is not targeted for the simulation of HVS *color processing*, it mainly operates on the achromatic channel only. Therefore it does not account for chromatic adaptation, color aftereffects and other phenomena of color vision; but we believe those can be pertinently included, if necessary.

The model assumes to input a calibrated HDR image and by modeling of the HVS features it is accordingly able to perform the *HDR tone mapping* task (for a calibrated HDR image). However, as the primary goal of the model is the correct simulation of the HVS contrast processing, the results for some extremely high dynamic range or not calibrated images can not outperform the results of specifically tuned tone mapping operators. Note however, that the HVS is also unable to see all the details in the scene simultaneously for extremely high dynamic ranges. From this point of view, the results of many "successful" tone mapping operators are not perceptually correct, as indicated by recent experimental studies [AFR*07, ČWNA08].

# Conclusions and Future Work

This dissertation concludes with this final chapter. Section 6.1 summarizes the thesis and gives some concluding remarks on my main contributions – contrast prescription, contrast-enhanced compression of synthetic content, and simulation of maladaptation in contrast domain, whereas Section 6.2 presents a short discussion of proposed methods and their possible extensions in future work.

## 6.1 Conclusions

The main motivation behind this thesis was to explore the use of perception-based contrast models in a computer graphics context. Such models are commonly found in lossy compression methods, where they "focus" the encoding process on perceptually significant data. However, despite their strong application to the compression, embedding visual models to steer computer graphics algorithms is not a common practice so far. This is surprising, since the results of such algorithms, usually in the form of images or animations, are intended for human observers. This thesis addresses this issue and takes a step towards the use of perceived contrast models to improve the visual quality and computational efficiency in important computer graphics applications. Specifically, I have considered multi-scale image enhancement and manipulation, compression of depth and motion video streams, and realistic reproduction of the visibility of photometric images and movies. To satisfy thesis requirements defined in Section 1.2, in each of these cases I have used a specific contrast model, and demonstrated its advantages in terms of extending the functionality of state-of-the-art, and improving application performance.

Chapter 3 presents an analysis of multi-scale image representations with respect to local and band-selective contrast manipulation in images. The results of this joint work with co-authors [PvA*10b] have shown that each band receives a significant portion of energy from its neighbors. Such behavior, referred to as the "energy leakage", results from a weak band separation of Gaussian-like filters, and while it allows to avoid ringing artifacts, it also becomes a serious drawback in contrast editing applications. Because the modification

of one band inherently affects the others, the user is forced to perform a single manipulation in multiple iterations, which reduces the practical value of such frameworks. My individual contribution to this work is a novel method that mitigates the above problem and improves the usability of multi-scale contrast editing applications. I have proposed a simple, yet efficient algorithm that relies on a Peli contrast model and restores the visibility of unmodified sub-band contrast. Since the energy leakage is common to all existing decomposition frameworks, I have shown how three state-of-the-art methods can be extended with contrast prescription method. Their performance was evaluated in several image editing scenarios, for which I have developed a local contrast manipulation application. The resulting software implementation, thanks to GPU acceleration, provides a real-time feedback and interaction even for decomposition methods involving PDE solvers.

Chapter 4 introduced a novel solution to a problem of efficient remote rendering of 3D content. In naïve approaches, the server sends a full-resolution video stream to the client whose only role is to decode and display the video. Together with co-authors [PvA*10b], we proposed a more balanced setup, where the server renders a sparse frame representation along with some auxiliary information that can be used in full-resolution frame reconstruction directly on the client side. As a result, by shifting some computations to client GPUs, our method better balances the server/client workload, allowing the server to handle more clients given the same amount of computational resources. At the same time, we showed that our approach delivers visual quality similar to naïve streaming at a given bandwidth limit. My main contribution to this work, which made all of the above possible, is design and implementation of auxiliary video stream compression. The auxiliary stream consists of per frame depth and motion information, which we concluded to be crucial to increasing the robustness of transmission and image reconstruction. I have determined that depth and motion share basic characteristics, i.e., they are mostly smooth and can be efficiently represented with edges. This key insight allowed me to develop a novel depth and motion video encoding scheme that compression-wise outperforms current state-of-the-art encoding standards, such as H.264. The compression method relies on an inherent relation between depth and underlying contrast in rendered images. To capture this relation, I have derived a perceptual contrast model that estimates the visual importance of depth edges. By precisely controlling the amount of encoded edges, the proposed method is capable of bandwidth and quality scalability, which is a necessary feature in network environments. I have also shown how depth and motion frames can be quickly reconstructed from edge representation. To enable fast reconstruction, I have developed a fast and GPU-friendly algorithm that provides a high quality approximation of PDE-based diffusion. The performance of the proposed system has been demonstrated and evaluated on an example of remote rasterization with complex shading. Our implementation of both server and client sides proved the method to be efficient enough to support real-time interaction. Additionally, we showed that depth and motion information on the client-side

allows for stereo generation and other attractive applications.

Chapter 5 focuses on modeling of maladaptation effect – a local and temporal loss of contrast sensitivity. Although this phenomena is usually fast and does not significantly affect our everyday functioning, its analysis and simulation might be beneficial in, for example, vehicle or aircraft control. For such a time-critical environments, accurate prediction of visibility of control equipment allows to improve the safety and responsiveness in varying illumination conditions. The result of a collaboration with co-authors [PvA*10a] is a system that renders a series of images of a scene as it would be seen by a maladapted eye over time. My contribution to this work is twofold. First, to enable accurate visualization of maladaptation, I have derived a novel HVS model that simulates the perception of contrast under dynamically changing lighting conditions. The time course of adaptation is modeled by considering both neural mechanisms, and pigment bleaching and regeneration. The proposed framework operates in multi-scale contrast domain and models supra-threshold effects like visual masking, while also accounting for contrast sensitivity and luminance (mal)adaptation. The simulation of maladaptation, as opposed to existing methods, is local, and models the shifting of peak spatial frequency sensitivity in maladapted vision in addition to the uniform decrease in contrast sensitivity among all frequencies. Additionally, I have developed a proof-of-concept software that is capable of processing HDR images and movies, and utilizes the GPU to visualize the maladaptation effect at interactive rates.

With the above conclusions, I have fulfilled both primary and secondary objectives of this thesis, as described in the introduction Section 1.2. Specifically, in each of the problems presented in Chapters 3–5 the solution comprised a perceptual contrast model that was essential to the corresponding application performance. This validates my initial claim that such models:

- allow for an implementation of contrast prescription in multi-scale image decomposition frameworks (Chapter 3),
- improve encoding performance and bandwidth scalability in depth and motion video compression (Chapter 4).
- enable the simulation of visual maladaptaion effect (Chapter 5),

These claims constitute the thesis, therefore by validating them I unambiguously proved the thesis.

Following the development of computer graphics and vision, one can observe a strong correlation between the complexity of proposed solutions and computational capabilities of hardware existing at the time. This relation is a result of disparity between insufficient hardware performance and high requirements of computer graphics and vision methods, which often have to be interactive and provide a real-time feedback. Consequently, the use

of complex and non-linear models of human vision was not feasible in the past. Nowadays, however, thanks to recent advances in high-performance computing and introduction of programmable graphics hardware (GPUs), the computation cost is less of an issue. In this thesis I showed that exploiting perceptual contrast models is beneficial for processing of visual data and the resulting benefits outweigh the cost of increased complexity. Given that such data is intended for human observers, the above statement might seem obvious, however, I have presented a solid scientific evidence to support this claim, and demonstrated its performance in solutions of computer graphics related problems.

## 6.2 Future Work

The following section discusses possible extensions to the work presented in Chapters 3–5.

Chapter 3 introduces the idea of contrast prescription – a method that allows the user to "lock" the appearance of sub-band contrast and reduce the effects of cross-band energy leakage. At the core of contrast restoration algorithm, I use a relatively simple multi-scale contrast metric that assumes the observer is adapted to a constant luminance value. This is a valid assumption for an ordinary LCD display, for which the adaptation is roughly equal to $100 cd/m^2$ and does not affect the perception of contrast. But when considering very bright light sources, such as HDR capable displays, our adaptation state varies spatially, which affects local contrast visibility thresholds. Consequently, in such scenarios the proposed method tends to overestimate contrast restoration multipliers for dark regions of the image. Conversely, for bright image regions the algorithm is not be able to compensate for the energy leakage. To address this issue and enable editing of HDR images directly on HDR-class displays, the prescription algorithm could be extended to model the apparent contrast change. A possible solution would involve a non-linear transducer function that takes into account the local adaptation state. To provide the user with a tool that modifies the contrast in a perceptually uniform way, such transducer would have to consider the effects of inter-channel contrast masking, since they become more apparent for HDR content.

In Chapter 4 we presented a client-server architecture for rendering and distribution of 3D content. The proposed system comprises many components, but despite all its advantages, there is still much room for improvements. In addition to the limitations discussed in Section 4.7, I find several directions with a particular prospect for future work. For instance, depth and motion video encoding can be significantly improved by considering more advanced methods for prediction of edge values. Currently, we compute the residual by subtracting edge values from their spatial or temporal neighbors with the same "laplacian sign". For simple edge structures, this produces residuals close to zero, however, in

case of scenes with high geometry complexity, where multiple edges overlap, a sign-based selection of a predictor might be ill-posed. The entire encoding algorithm could be adapted to the compression of data different than depth, such as HDR videos. Many HDR compression techniques separate frame data into base HDR illumination and LDR contrast enhancement layers. The illumination layer shares many characteristics with depth data, i.e, it is mostly smooth and contains sharp edges, which suggest that the edge-based encoding could be used in its compression. Another track of future research is related to the scalability of the proposed architecture in various graphics pipelines. The performance of our framework has been evaluated on an example of rasterization engine fitted with complex shading methods. However, the application of proposed method is not confined to rasterization techniques. In fact, I believe that it can be even more beneficial for ray-tracing, where frames are costly to compute and require high-performance computers, or even a cluster. Since our scheme reduces the rendering cost of a single frame, the resulting spare resources could be used to increase either the framerate or the number of clients per server.

Chapter 5 proposes a perceptual contrast model designed to simulate the effect of visual maladaptation. Since the treatment of color in our work is limited, a straightforward extension to the model is to account for adaptation mechanisms in chromatic channels. The simulation of adaptation in all color channels would allow to visualize color aftereffects and other phenomena of color vision. A more challenging problem is to extend the proposed method with, so called, *temporal contrast sensitivity function* that simulates human sensitivity to luminance variations over time. Such mechanism could be beneficial for visualization of HDR movies, where luminance patterns change frequently. Additionally, the plausibility of the model can be improved with the simulation of glare effects. Particularly interesting would be to simulate the glare in maladapted states, when the pupil diameter changes dynamically.

At present I am pursuing research which extends the depth encoding method presented in Chapter 4. In particular, I am interested in a relation between depth and luminance contrast in a context of video compression of stereo content. When we look at objects in the world, their corresponding "depth" maps to binocular disparity – a difference in image location of an object seen by the left and right eyes. Similar to luminance contrast, disparity perception finds its origin in inherent limitations of human visual system. For instance, the visibility of disparity is reduced in a presence of another disparity stimuli or if a contrast of underlying luminance is low. I would like to exploit these limitations to improve on the video coding efficiency, while still maintaining high visual quality of stereo content.

# High Performance Image Processing

The following appendix highlights some of the technical contributions made during the course of work on this thesis. Specifically, we focus on an HDR image processing pipeline, where data is represented using floating-point numbers. Such a precision promises high-quality results, but also leads to a significant computational overhead, and therefore, to ensure high performance of applications, an efficient hardware and algorithmic acceleration is necessary. Here we present a holistic approach to image processing, where algorithms are optimized in two ways. First, we design our framework to exploit multi-level memory hierarchy and multiprocessing capabilities of computing hardware, both of which are common to current high-performance architectures. Next, we conform to a particular computing platform, and provide optimized variants of basic image processing operations. For example, our CPU-based implementation uses SIMD floating-point units to exploit data level parallelism characteristic to many image processing methods. Although here we show results for regular CPUs, the proposed approach maps well to heterogeneous computing environments, such as OpenGL/CL or CUDA, and as has been shown in previous chapters, significantly improves the application performance.

## A.1 HDRI Acceleration Pipeline

Compared to regular LDR images, HDR data require more effort to process and display simply because we work on floating-point values instead of integers, and the data represents photometric quantities rather than display-referred pixel intensities. In addition to increased computational demands, HDR images occupy more memory, which further reduces the execution speed, especially for image processing methods whose performance is bound by system memory bandwidth. In such cases, as soon as the image data becomes larger than the cache memory, the performance will degrade by an order of magnitude. In general the above situation is referred to as the *memory wall* [RKB*09] and results from a disparity of speed between the processor and the system memory. HDR images, due to their increased size, are strongly affected by the memory wall problem.

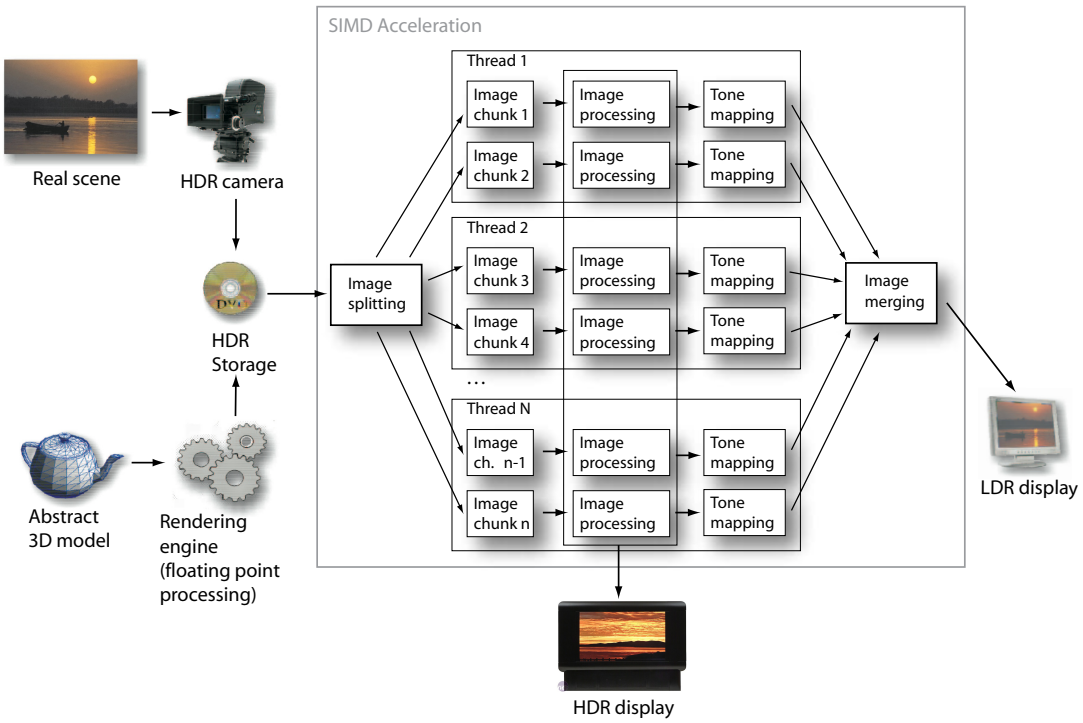


**Figure A.1** – A high-level view of the proposed image processing pipeline

*The high-performance of our framework results from an extensive use of parallel computation and increased data locality. The input image is divided into logical tiles that fit in the cache memory and can be processed independently by available compute threads. Each thread executes on his tile a set of user-scheduled image processing operations. Since tiles are small, the data does not leave the cache between operations, which mitigates the memory wall problem and significantly improves the performance.*

We address this issue in [MP08], where we presented an image processing framework optimized for HDR content. As shown in Figure A.1, we alleviate the memory wall problem by implementing a form of *cache blocking* [HP07], which significantly reduces the cache miss rate, and improves data reuse between subsequent calls to primitive image processing functions.

The basic functionality of the proposed framework includes matrix operations, 1D/2D convolution, accumulation operations, conditional writes, etc. Additionally, it offers HDR-specific functions, such as tone-mapping. To fully exploit computing capacity of current processors, we optimized the architecture for parallel and SIMD hardware. In some cases, we sacrificed the accuracy of computation in order to improve the parallelism and simplify the vectorization. This was motivated by the fact that, in contrast to scientific computing, computer graphics applications are tolerant to approximations. For example, to compute $\log_2(x)$, we use features specific to a single precision floating-point number representation (float). As described in IEEE 754-1985 specification, a single precision float is defined as $(-1)^s 2^e m$, where $s$ is a sign bit, $e$ is an 8-bit exponent and $m$ is a 24-bit normalized mantissa

| Operation | Time [ms] | | | Speed-up |
|---|---|---|---|---|
| | FPU/st | SIMD/st | SIMD/mt | |
| IMAGE BLENDING | 257 | 238 | 139 | 1.85 |
| XYZ TO RGB CONVERSION | 64 | 38 | 20 | 3.2 |
| GEOMETRIC MEAN | 395 | 33 | 19 | 20.79 |
| GAMMA CORRECTION | 1971 | 302 | 157 | 12.55 |
| GAMMA TMO | 3356 | 621 | 375 | 8.95 |
| PHOTOGRAPHIC TMO [RSSF02] | 3146 | 590 | 360 | 8.74 |

**Table A.1** – Low-level image processing timings for different code-paths

*Here we show timings for internal functions of CPU-based implementation of the proposed framework. To emphasize the importance of low-level optimization of the pipeline, we show results for 3 implementations: FPU/st (single-threaded, c++), SIMD/st (single-threaded, SIMD assembly) and SIMD/mt (multi-threaded, SIMD assembly). The speed-up corresponds to a ratio between FPU and multi-threaded SIMD versions.*

($m \in (1,2)$). Therefore, $\log_2(x)$, where $x > 0$, can be rewritten as $\log_2(x) = \log_2(2^e m) = e + \log_2(m)$, and to compute its value we simply sum the exponent with an approximation of $\log_2(m)$. Since the range of mantissa is small, we can precisely approximate $\log_2(m)$ with polynomials. In our implementation we use a fifth degree Chebyshev polynomial, which generates small relative error ($10^{-6}$) and can be computed efficiently with SIMD approach.

The performance of such optimizations for various implementations is demonstrated in Table A.1. To produce the timings we used 6Mpixels HDR images and a mainstream PC equipped with AMD Athlon X2 3800+ dual-core CPU and 2GB DDR RAM. For a more detailed discussion of the results we refer the reader to the original paper [MP08].

## A.2 Stencil Computations

In stencil computations, a small neighborhood of each pixel in an image contributes to the new value of this pixel. In the simplest case we have constant weights which are multiplied with the neighbor pixel and summed up, e.g. in 2D a general 5-point stencil has a weight for the element itself and 4 weights for each of its direct neighbors in the 2 spatial dimensions x and y:

$$v'(x,y) = w_1 v(x,y) + w_2 v(x-1,y) + w_3 v(x+1,y) + w_4 v(x,y-1) + w_5 v(x,y+1).$$

This computational pattern occurs very often in computer graphics because discretized differential operators take this form (with possibly spatially varying weights $w_k(x,y)$). They

can be found in (but not limited to) edge-preserving decompositions [FFLS08], gradient domain tone-mapping [MMS06], edge-detection [Can86] and laplacian pyramid [BA83]. In this thesis we also make use of such patterns, particularly in decomposition frameworks from Chapter 3 and edge-preserving diffusion in Chapter 4.

A big problem with these operations is the low arithmetic intensity, performing just one multiply-and-add operation for each retrieval of one neighbor element [SSP11a]. The cache helps in this scenario to retrieve the x-neighbors $v(x-1,y), v(x+1,y)$ (unit stride) quickly. Assuming the cache is large enough to fit several rows of the image data, the y-neighbors $v(x,y-1), v(x,y+1)$ can also be fetched from cache after reading them once. From there on, however, the execution time does not improve if we make the cache even bigger, because no additional data reuse can occur. Only if the cache size becomes large enough to hold the entire image, then we could benefit in multiple applications of the stencil, which would all happen in cache. With typical cache and image sizes, we find ourselves in the situation that the neighboring row fit, but the image is much larger than the cache. So all neighbors come from the cache, but every element $v(x,y)$ has to be read once from system memory for every application of the stencil.

So the achieved execution time for the entire image remains restricted by the system bandwidth [SSP11a]. This fact is particularly painful if the stencil application is repeated hundreds of times, as happens in iterative solvers of linear equation systems, or the discrete time evolution in partial differential equations. In each iteration the entire domain has to be fetched from system memory in the naive implementation. We cannot solve the problem with more advanced cache hardware.

The key idea in efficient iterative stencil computations is to perform multiple iterations locally on small tiles of the image. Then data in the cache can be reused for multiple iterations without going to the system memory, thus mitigating the memory wall problem. Because of the dependence on neighbors in each iteration step, multiple local iterations enforce index dependencies in such computations. In [SSPS11] we developed a novel scheme, referred to as *cache accurate time skewing*, that follows such computation pattern. Because more data is read from the cache, the cache bandwidth becomes relevant for the execution time of this technique. Our implementation on multi-core SIMD CPUs performs beyond system bandwidth limitations as though gigabytes of data could reside in an enormous on-chip cache.

In [SSPS10] we investigated, so called, *cache oblivious* schemes that dynamically adapt tile sizes to exploit multiple levels of cache memory in current CPUs. Our novel contribution in this work is a cache oblivious method based on parallelogram shaped tiles. Its performance stems from a tiling structure that caters for data locality, parallelism and vectorization simultaneously. Rather than tiling the iteration space from inside, we take an exterior approach with a predefined hierarchy, simple regular parallelogram tiles and

a locality preserving parallelization. These advantages come at the cost of an irregular work-load distribution but a tightly integrated load-balancer ensures a high utilization of all resources.

# Bibliography (Own work)

[MP08] Mantiuk R., Pająk D.: Acceleration of high dynamic range imaging pipeline based on multi-threading and SIMD technologies. In *Lecture Notes in Computer Science (Proceedings of International Conference on Computational Science)* (2008), pp. 780–789.

[PHE*11] Pająk D., Herzog R., Eisemann E., Myszkowski K., Seidel H.-P.: Scalable remote rendering with depth and motion-flow augmented streaming. *Computer Graphics Forum (Proceedings of Eurographics) 30*, 2 (2011), 415–424.

[PvA*10a] Pająk D., Čadík M., Aydin T. O., Myszkowski K., Seidel H.-P.: Visual maladaptation in contrast domain. In *Proceedings of IS&T/SPIE's Human Vision and Electronic Imaging* (San Jose, CA, USA, 2010), Society of Photo-Optical Instrumentation Engineers, pp. 1–12.

[PvA*10b] Pająk D., Čadík M., Aydin T. O., Okabe M., Myszkowski K., Seidel H.-P.: Contrast prescription for multiscale image editing. *The Visual Computer Journal 26* (2010), 739–748.

[SSP11a] Strzodka R., Shaheen M., Pająk D.: Impact of system and cache bandwidth on stencil computation across multiple processor generations. In *Proceedings of 2nd Workshop on Applications for Multi and Many Core Processors* (2011).

[SSP11b] Strzodka R., Shaheen M., Pająk D.: Time skewing made simple. In *Proceedings of the 16th ACM Symposium on Principles and Practice of Parallel Programming* (New York, NY, USA, 2011), PPoPP 2011, ACM, pp. 295–296.

[SSPS09] Strzodka R., Shaheen M., Pająk D., Seidel H.-P.: Overcoming bandwidth limitations in visual computing. In *Proceedings of Intel Visual Computing Research Conference* (2009).

[SSPS10] Strzodka R., Shaheen M., Pająk D., Seidel H.-P.: Cache oblivious parallelograms in iterative stencil computations. In *ICS 2010: Proceedings*

*of the 24th ACM International Conference on Supercomputing* (June 2010), ACM, pp. 49–59.

[SSPS11] Strzodka R., Shaheen M., Pająk D., Seidel H.-P.: Cache accurate time skewing in iterative stencil computations. In *Proceedings of the International Conference on Parallel Processing (ICPP)* (Sept. 2011), IEEE Computer Society, pp. 571–581.

# Bibliography

[ABF*04] Alexander K. R., Barnes C. S., Fishman G. A., Pokorny J., Smith V. C.: Contrast Sensitivity Deficits in Inferred Magnocellular and Parvocellular Pathways in Retinitis Pigmentosa. *Investigative Ophthalmology & Visual Science 45*, 12 (2004), 4510–4519. 90, 91

[AFR*07] Akyüz A. O., Fleming R., Riecke B. E., Reinhard E., Bülthoff H. H.: Do HDR Displays Support LDR Content? A Psychophysical Evaluation. *Proceedings of SIGGRAPH 26*, 3 (2007), 38. 103

[AMHH08] Akenine-Möller T., Haines E., Hoffman N.: *Real-Time Rendering 3rd Edition*. A. K. Peters, Ltd., Natick, MA, USA, 2008. 10

[AMS09] Aydin T. O., Mantiuk R., Seidel H.-P.: Predicting display visibility under dynamically changing lighting conditions. *Computer Graphics Forum (Proceedings of Eurographics) 28*, 3 (2009). 90, 91, 95

[AvMS10] Aydin T. O., Čadík M., Myszkowski K., Seidel H.-P.: Video quality assessment for computer graphics applications. *ACM Transactions on Graphics 29* (Dec 2010), 161:1–161:12. 42

[BA83] Burt P. J., Adelson E. H.: The laplacian pyramid as a compact image code. *IEEE Transactions on Communications 31* (1983), 532–540. 7, 29, 32, 35, 51, 92, 114

[Bar99] Barten P. G.: Contrast sensitivity of the human eye and its effects on image quality. In *Proceedings of IS&T/SPIE's Human Vision and Electronic Imaging* (1999), Society of Photo-Optical Instrumentation Engineers. 1, 23, 90

[BGG09] Bernabé G., García J. M., González J.: A lossy 3d wavelet transform for high-quality compression of medical video. *Journal of Systems and Software 82* (Mar 2009), 526–534. 42

[BPD06] Bae S., Paris S., Durand F.: Two-scale tone management for photographic look. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* (2006), pp. 637–645. 51

[BSMH98] Black M., Sapiro G., Marimont D., Heeger D.: Robust anisotropic diffusion. *IEEE Transactions on Image Processing 7*, 3 (Mar 1998), 421–432. 51

[Can86] Canny J.: A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence 8* (November 1986), 679–698. 114

[CBPZ04] Cheng L., Bhushan A., Pajarola R., Zarki M. E.: Real-time 3D graphics streaming using MPEG-4. In *Proceedings IEEE/ACM Workshop on Broadboand Wireless Services and Applications* (2004). 68

[COMF99] Cohen-Or D., Mann Y., Fleishman S.: Deep compression for streaming texture intensive animations. In *Proceedings of SIGGRAPH* (1999), pp. 261–268. 67

[CPD07] Chen J., Paris S., Durand F.: Real-time edge-aware image processing with the bilateral grid. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* (2007), p. 103. 51

[ČWNA08] Čadík M., Wimmer M., Neumann L., Artusi A.: Evaluation of HDR tone mapping methods using essential perceptual attributes. *Computers & Graphics 32* (2008), 330–349. 103

[Dal93] Daly S.: The visible differences predictor: An algorithm for the assessment of image fidelity. In *Digital Images and Human Vision* (1993), Watson A. B., (Ed.), MIT Press, pp. 179– 206. 20, 23, 24, 29, 91, 94, 96

[Dau85] Daugman J. G.: Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *Journal of the Optical Society of America A 2*, 7 (Jul 1985), 1160–1169. 29

[DD02] Durand F., Dorsey J.: Fast bilateral filtering for the display of high-dynamic-range images. In *Proceedings of SIGGRAPH* (2002), ACM Press, pp. 257–266. 51

[DER*10] Didyk P., Eisemann E., Ritschel T., Myszkowski K., Seidel H.-P.: Perceptually-motivated real-time temporal upsampling of 3D content for high-refresh-rate displays. *Computer Graphics Forum (Proceedings of Eurographics) 29*, 2 (2010), 713–722. 78

[dir08] Dirac video compression specification, Sept 2008. http://diracvideo.org/. 42

[Dow87] Dowling J.: *The Retina: An Approachable Part of the Brain.* Harvard University Press, 1987. 90

[DRE*10] Didyk P., Ritschel T., Eisemann E., Myszkowski K., Seidel H.-P.: Adaptive image-space stereo view synthesis. In *Proceedings of 15th International Workshop on Vision, Modeling and Visualization* (2010). 78

[ED04] Eisemann E., Durand F.: Flash photography enhancement via intrinsic relighting. *ACM Transactions on Graphics (Proceedings SIGGRAPH) 23*, 3 (2004), 673–678. 51

[Eld99] Elder J. H.: Are edges incomplete? *International Journal of Computer Vision 34*, 2-3 (Aug 1999), 97–122. 68

[EWG99] Eisert P., Wiegand T., Girod B.: Rate-distortion-efficient video compression using a 3D head model. In *International Conference on Image Processing* (1999), pp. 217–221. 68

[Fai05] Fairchild M.: *Color Appearance Models*, 2 ed. Addison-Wesley, 2005. 13, 16

[FAR07] Fattal R., Agrawala M., Rusinkiewicz S.: Multiscale shape and detail enhancement from multi-light image collections. *ACM Transactions on Graphics (Proceedings SIGGRAPH) 26*, 3 (Aug 2007). 51

[Fat09] Fattal R.: Edge-avoiding wavelets and their applications. *ACM Transactions on Graphics 28*, 3 (Jul 2009), 22:1–22:10. 7, 29, 35, 51, 53, 58, 82

[FE09] Fechteler P., Eisert P.: Depth map enhanced macroblock partitioning for H.264 video coding of computer graphics content. In *International Conference on Image Processing* (2009), pp. 3441–3444. 68

[Feh04] Fehn C.: Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV. In *Stereoscopic Displays and Virtual Reality Systems XI* (2004), vol. 5291, Society of Photo-Optical Instrumentation Engineers, pp. 93–104. 68

[FFLS08] Farbman Z., Fattal R., Lischinski D., Szeliski R.: Edge-preserving decompositions for multi-scale tone and detail manipulation. *ACM Transactions on Graphics 27*, 3 (2008), 67:1–67:10. 29, 34, 35, 51, 82, 114

[Fol94] Foley J. M.: Human luminance pattern-vision mechanisms: masking experiments require a new model. *Journal of the Optical Society of America A 11*, 6 (Jun 1994), 1710–1719. 27

[FPSG96] Ferwerda J. A., Pattanaik S. N., Shirley P., Greenberg D. P.: A model of visual adaptation for realistic image synthesis. In *Proceedings of SIGGRAPH* (New York, NY, USA, 1996), ACM, pp. 249–258. 11, 89

[FPSG97] Ferwerda J. A., Pattanaik S. N., Shirley P. S., Greenberg D. P.: A model of visual masking for computer graphics. In *Proceedings of SIGGRAPH* (Aug 1997), pp. 143–152. 27, 88

[GB00] Gibson J. D., Bovik A. (Eds.): *Handbook of Image and Video Processing*. Academic Press, Inc., Orlando, FL, USA, 2000. 33, 57

[GGSC96] Gortler S. J., Grzeszczuk R., Szeliski R., Cohen M. F.: The lumigraph. In *Proceedings of SIGGRAPH* (1996), ACM, pp. 43–54. 71

[GS75] Georgeson M. A., Sullivan G. D.: Contrast constancy: deblurring in human vision by spatial frequency channels. *The Journal of Physiology 252*, 3 (1975), 627–656. 23

[GW02] Gonzalez R. C., Woods R. E.: *Digital Image Processing*, 2nd ed. Prentice-Hall, New Jersey, 2002. 30, 32, 51

[GWW*08] Galic I., Weickert J., Welk M., Bruhn A., Belyaev A. G., Seidel H.-P.: Image compression with anisotropic diffusion. *Journal of Mathematical Imaging and Vision 31*, 2-3 (2008), 255–269. 68

[Hec01] Hecht E.: *Optics*, 4 ed. Addison Wesley, Aug 2001. 14

[HEMS10] Herzog R., Eisemann E., Myszkowski K., Seidel H.-P.: Spatio-temporal upsampling on the GPU. In *Proceedings of I3D* (2010), ACM, pp. 91–98. 67, 73

[HG95] Hahn L. W., Geisler W. S.: Adaptation mechanisms in spatial vision–i. bleaches and backgrounds. *Vision Research 35*, 11 (1995), 1585 – 1594. 90, 91, 95

[HKMS08] Herzog R., Kinuwaki S., Myszkowski K., Seidel H.-P.: Render2MPEG: A perception-based framework towards integrating rendering and video compression. *Computer Graphics Forum (Proceedings of Eurographics) 27*, 2 (2008), 183–192. 68

[Hoo98] Hood D.: Lower-level visual processing and models of light adaptation. *Annual Review of Psychology 49* (1998), 503–535. 20

[HP07] Hennessy J. L., Patterson D. A.: *Computer Architecture - A Quantitative Approach*, 4th ed. Morgan Kaufmann, 2007. 112

[Hun95] Hunt R. W. G.: *The reproduction of colour*, 5 ed. Fountain Press, 1995. 89, 93

[IFM05] Irawan P., Ferwerda J. A., Marschner S. R.: Perceptually based tone mapping of high dynamic range image streams. In *Computer Graphics Forum (Proceedings of EGSR)* (2005), pp. 231–242. 21, 88, 89, 93, 97, 98

[ITU99] ITU-T: Information technology – coded representation of picture and audio information – lossy/lossless coding of bi-level images, itu-t recommendation t.82 – iso/iec international standard 14492 final committee draft. 75

[JO05] Jansen M. H., Oonincx P. J.: *Second Generation Wavelets and Applications*. Springer, 2005. 42, 58

[JSMB*92] Jonas J. B., Schmidt A. M., Müller-Bergh J. A., Schlötzer-Schrehardt U. M., Naumann G. O.: Human optic nerve fiber count and optic disc size. *Investigative Ophthalmology & Visual Science 33*, 6 (1992), 2012–8. 15

[JWB04] Jiang Z., Wong T.-T., Bao H.: Depth image sequences compression using programmable graphics hardware. TR-2004-01, 2004. 68

[KAF*07] Kauff P., Atzpadin N., Fehn C., Müller M., Schreer O., Smolic A., Tanger R.: Depth map creation and image-based rendering for advanced 3dtv services providing interoperability and scalability. *Signal Processing: Image Communication 22*, 2 (2007), 217 – 234. Special issue on 3D video and television. 68

[KbCTS01] Krishnamurthy R., bing Chai B., Tao H., Sethuraman S.: Compression and transmission of depth maps for image-based rendering. In *International Conference on Image Processing* (2001), pp. 828–831. 68

[Kel74] Kelly D. H.: Spatio-temporal frequency characteristics of color-vision mechanisms. *Journal of the Optical Society of America (1917-1983) 64* (Jul 1974). 90

[Kel83] Kelly D. H.: Spatiotemporal variation of chromatic and achromatic contrast thresholds. *Journal of the Optical Society of America 73*, 6 (Jun 1983), 742–749. 23

[KMS07] Krawczyk G., Myszkowski K., Seidel H.-P.: Contrast restoration by adaptive countershading. *Computer Graphics Forum (Proceedings of Eurographics) 26*, 3 (2007), 581–590. 58

[KSY92] Krantz J. H., Silverstein L. D., Yeh Y.-Y.: Visibility of transmissive liquid crystal displays under dynamic lighting conditions. *Human Factors 34*, 5 (1992), 615–632. 87

[LAA08] Li Y., Adelson E., Agarwala A.: Scribbleboost: Adding classification to edge-aware interpolation of local image and video adjustments. *Computer Graphics Forum 27*, 4 (2008), 1255–1264. 51

[LBB88] Lagendijk R., Biemond J., Boekee D.: Regularized iterative image restoration with ringing reduction. *IEEE Transactions on Acoustics, Speech and Signal Processing 36*, 12 (Dec 1988), 1874–1888. 51

[LCD06] Luft T., Colditz C., Deussen O.: Image enhancement by unsharp masking the depth buffer. *ACM Transactions on Graphics (Proceedings of SIGGRAPH) 25*, 3 (2006), 1206–1213. 78

[Len93] Lennie P.: Roles of M and P pathways. In *Contrast Sensitivity*, Shapley R., Lam D. M.-K., (Eds.). MIT Press, 1993, ch. 11, pp. 201– 213. 90

[Lev95] Levoy M.: Polygon-assisted JPEG and MPEG compression of synthetic images. In *Proceedings of SIGGRAPH* (1995), pp. 21–28. 67

[LF80] Legge G. E., Foley J. M.: Contrast masking in human vision. *Journal of the Optical Society of America 70*, 12 (Dec 1980), 1458–1471. 26, 27, 28

[LFUS06] Lischinski D., Farbman Z., Uyttendaele M., Szeliski R.: Interactive local adjustment of tonal values. *ACM Transactions on Graphics 25*, 3 (2006), 646–653. 53, 59

[Lin98] Lindeberg T.: Feature detection with automatic scale selection. *International Journal of Computer Vision 30* (1998), 79–116. 29

[Liv02] Livingstone M.: *Vision and Art: The Biology of Seeing*. Harry N. Abrams, 2002. 58

[LLW04] Levin A., Lischinski D., Weiss Y.: Colorization using optimization. *ACM Transactions on Graphics (Proceedings of SIGGRAPH) 23*, 3 (2004), 689–694. 51

[LOL97] Leduc J.-P., Odobez J.-M., Labit C.: Adaptive motion-compensated wavelet filtering for image sequence coding. *IEEE Transactions on Image Processing 6*, 6 (1997), 862–878. 42

[LPS03] Leonova A., Pokorny J., Smith V. C.: Spatial frequency processing in inferred pc- and mc-pathways. *Vision Research 43*, 20 (2003), 2133 – 2139. 90, 91

[LRP97] Larson G. W., Rushmeier H., Piatko C.: A visibility matching tone reproduction operator for high dynamic range scenes. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 3* (1997), 291–306. 89, 93

[LSA05] Li Y., Sharan L., Adelson E. H.: Compressing and companding high dynamic range images with subband architectures. *ACM Transactions on Graphics (Proceedings SIGGRAPH) 24*, 3 (2005), 836–844. 51

[LSTS04] Li Y., Sun J., Tang C.-K., Shum H.-Y.: Lazy snapping. *ACM Transactions on Graphics (Proceedings of SIGGRAPH) 23*, 3 (2004), 303–308. 51

[MD08] Maitre M., Do M.: Joint encoding of the depth image based representation using shape-adaptive wavelets. In *International Conference on Image Processing* (2008), pp. 1768–1771. 68

[MDMS05] Mantiuk R., Daly S., Myszkowski K., Seidel H.-P.: Predicting visible differences in high dynamic range images – model and its calibration. In *Proceedings of IS&T/SPIE's Human Vision and Electronic Imaging* (2005), vol. 5666, pp. 204–214. 91, 94

[MF98] Murad A., Fuja T.: Exploiting the residual redundancy in motion estimation vectors to improve the quality of compressed video transmitted over noisy channels. In *International Conference on Image Processing* (1998), pp. 497–501. 68

[MH80] Marr D., Hildreth E.: Theory of edge detection. *Proceedings of the Royal Society of London. Series B, Biological Sciences 207*, 1167 (1980), 187–217. 7, 29, 31, 53

[MKRH11] Mantiuk R., Kim K. J., Rempel A. G., Heidrich W.: Hdr-vdp-2: a calibrated visual metric for visibility and quality predictions in all luminance conditions. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* (New York, NY, USA, 2011), ACM, pp. 40:1–40:14. 24

[MMS06] Mantiuk R., Myszkowski K., Seidel H.-P.: A perceptual framework for contrast processing of high dynamic range images. *ACM Transactions on Applied Perception 3*, 3 (2006), 286–308. V, 7, 35, 51, 58, 63, 90, 114

[MMS*09] Merkle P., Morvan Y., Smolic A., Farin D., Müller K., de With P. H. N., Wiegand T.: The effects of multiview depth video compression on multiview rendering. *Signal Processing: Image Communcation 24*, 1-2 (2009), 73–88. 68, 82, 83

[NDS*08] Nave I., David H., Shani A., Laikari A., Eisert P., Fechteler P.: Games@Large graphics streaming architecture. In *International Symposium on Consumer Electronics (ISCE)* (2008). 67

[NR66] Naka K. I., Rushton W. A. H.: S-potentials from luminosity units in the retina of fish (cyprinidae). *The Journal of Physiology 185* (Aug 1966), 587–599. 25

[OBW*08] Orzan A., Bousseau A., Winnemöller H., Barla P., Thollot J., Salesin D.: Diffusion curves: A vector representation for smooth-shaded images. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* (2008), vol. 27, pp. 92:1–92:8. 82

[Pae91] Paeth A. W.: Image file compression made easy. In *Graphic GEMS II* (1991). 77

[Pel90] Peli E.: Contrast in complex images. *Journal of the Optical Society of America A 7*, 10 (Oct 1990), 2032–2040. 52

[PF05] Pharr M., Fernando R.: *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation.* Addison-Wesley Professional, 2005. 10

[PFFG98] Pattanaik S. N., Ferwerda J. A., Fairchild M. D., Greenberg D. P.: A multiscale model of adaptation and spatial vision for realistic image display. In *Proceedings of SIGGRAPH* (1998), ACM Press, pp. 287–298. 16, 88, 89

[PJO*09] Park Y. K., Jung K., Oh Y., Lee S., Kim J. K., Lee G., Lee H., Yun K., Hur N., Kim J.: Depth-image-based rendering for 3DTV service over T-DMB. *Signal Processing: Image Communication 24*, 1-2 (2009), 122–136. 68

[PM90] Perona P., Malik J.: Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence 12*, 7 (Jul 1990), 629–639. 51

[PM92] Pennebaker W. B., Mitchell J. L.: *JPEG Still Image Data Compression Standard*, 1st ed. Kluwer Academic Publishers, Norwell, MA, USA, 1992. 35, 37, 41, 42, 43

[Poy98] Poynton C.: The rehabilitation of gamma. In *Proceedings of IS&T/SPIE's Human Vision and Electronic Imaging* (1998), Society of Photo-Optical Instrumentation Engineers. 18

[PTYG00] Pattanaik S. N., Tumblin J., Yee H., Greenberg D. P.: Time-dependent visual adaptation for fast realistic image display. In *Proceedings of SIGGRAPH* (2000), pp. 47–54. 88, 89

[RAGS01] Reinhard E., Ashikhmin M., Gooch B., Shirley P.: Color transfer between images. *IEEE Computer Graphics and Applications 21* (Sept 2001), 34–41. 16

[RHD*10] Reinhard E., Heidrich W., Debevec P., Pattanaik S., Ward G., Myszkowski K.: *High Dynamic Range Imaging*, 2 ed. Morgan Kaufmann, 2010. 17, 18, 19

[Ric10] Richardson I. E.: *The H.264 Advanced Video Compression Standard*, 2nd ed. John Wiley & Sons, Ltd, 2010. 35, 43

[RKB*09] Rogers B. M., Krishna A., Bell G. B., Vu K., Jiang X., Solihin Y.: Scaling the bandwidth wall: challenges in and avenues for cmp scaling. In *Proceedings of the 36th Annual International Symposium on Computer Architecture* (New York, NY, USA, 2009), ISCA '09, ACM, pp. 371–382. 111

[RSSF02] Reinhard E., Stark M., Shirley P., Ferwerda J.: Photographic tone reproduction for digital images. In *Proceedings of SIGGRAPH* (2002), ACM Press, pp. 267–276. 101, 113

[RTS*07] Rempel A. G., Trentacoste M., Seetzen H., Young H. D., Heidrich W., Whitehead L., Ward G.: Ldr2hdr: on-the-fly reverse tone mapping of legacy video and photographs. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* (New York, NY, USA, 2007), ACM, p. 39. 102

[SaLY*08] Sitthi-amorn P., Lawrence J., Yang L., Sander P. V., Nehab D.: An improved shading cache for modern GPUs. In *Proceedings of Graphics Hardware* (2008), pp. 95–101. 67, 69

[SCE01] Skodras A., Christopoulos C., Ebrahimi T.: The JPEG 2000 still image compression standard. *IEEE Signal Processing Magazine* (2001). 29

[Sil03] Silverstein L.: Display visibility in dynamic lighting environments: Impact on the design of portable and vehicular displays. In *Proceedings of International Display Manufacturing Conference* (Taipei, Taiwan, 2003). 87

[SLW*08] Stich T., Linz C., Wallraven C., Cunningham D., Magnor M.: Perception-motivated Interpolation of Image Sequences. In *Proceedings of Symposium for Applied Perception in Graphics and Visualization* (2008), pp. 97–106. 70

[SM10] Salomon D., Motta G.: *Handbook of Data Compression*, 5th ed. Springer, 2010. 43

[SS63] Stevens J. C., Stevens S. S.: Brightness function : Effects of adaptation. *Journal of the Optical Society of America 53*, 3 (Mar 1963), 375–385. 24

[SSD09] Subr K., Soler C., Durand F.: Edge-preserving multiscale image decomposition based on local extrema. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* (Dec 2009), Annual Conference Series, ACM, ACM Press. 51

[Swe97] Sweldens W.: The lifting scheme: A construction of second generation wavelets. *SIAM Journal on Mathematical Analysis 29*, 2 (1997), 511–546. 58

[Sze10] Szeliski R.: *Computer Vision: Algorithms and Applications*, 1st ed. Springer-Verlag New York, LLC, Nov 2010. 32

[TM98] Tomasi C., Manduchi R.: Bilateral filtering for gray and color images. In *Proceedings of the 6th International Conference on Computer Vision (ICCV 98)* (1998), IEEE Computer Society, p. 839. 34, 51

[TM01] Taubman D. S., Marcellin M. W.: *JPEG 2000: Image Compression Fundamentals, Standards and Practice*. Kluwer Academic Publishers, Norwell, MA, USA, 2001. 16, 35, 37, 42

[TT99] Tumblin J., Turk G.: LCIS: A boundary hierarchy for detail-preserving contrast reduction. In *Proceedings of SIGGRAPH* (1999), pp. 83–90. 51

[URB97] Uytterhoeven G., Roose D., Bultheel A.: Wavelet transforms using the lifting scheme, 1997. 29, 58

[Vit87] Vitter J. S.: Design and analysis of dynamic huffman codes. *Journal of ACM 34* (Oct 1987), 825–845. 46

[Wan95] Wandell B. A.: *Foundations of Vision*. Sinauer Associates, Inc., 1995. 16, 17

[War94] Ward G.: A contrast-based scalefactor for luminance display. *Graphics Gems IV* (1994), 415– 421. 89

[Wat87] Watson A. B.: The cortex transform: rapid computation of simulated neural images. *Computer Vision, Graphics, and Image Processing 39* (Sept 1987), 311–327. 29

[Wat94] Watson A. B.: Perceptual optimization of dct color quantization matrices. In *IEEE International Conference on Image Processing* (1994), pp. 100–104. 37, 43

[Wil80] Wilson H.: A transducer function for threshold and suprathreshold human vision. *Biological Cybernetics 38* (1980), 171–178. 96

[WKC94] Wallach D. S., Kunapalli S., Cohen M. F.: Accelerated MPEG compression of dynamic polygonal scenes. In *Proceedings of SIGGRAPH* (1994), pp. 193–197. 68

[WNC87] Witten I. H., Neal R. M., Cleary J. G.: Arithmetic coding for data compression. *Communications of the ACM 30*, 6 (1987), 520–540. 44, 75

[WS97] Watson A. B., Solomon J. A.: Model of visual contrast gain control and pattern masking. *Journal of the Optical Society of America A 14*, 9 (Sept 1997), 2379–2391. 26, 27

[WS00] Wyszecki G., Stiles W. S.: *Color Science: Concepts and Methods, Quantitative Data and Formulae (Wiley Series in Pure and Applied Optics)*, 2 ed. Wiley-Interscience, Aug 2000. 10

[WSBL03] Wiegand T., Sullivan G. J., Bjontegaard G., Luthra A.: Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology 13*, 7 (Jul 2003), 560–576. 36

[x26] x264 – an implementation of H.264 video coder. `http://www.videolan.org/developers/x264.html`. 83

[YMMS06] Yoshida A., Mantiuk R., Myszkowski K., Seidel H.-P.: Analysis of reproducing real-world appearance on displays of varying dynamic range. *Computer Graphics Forum (Proceedings of Eurographics) 25* (2006). 33

[YSL08] Yang L., Sander P. V., Lawrence J.: Geometry-aware framebuffer level of detail. *Computer Graphics Forum (Proceedings of EGSR) 27*, 4 (2008), 1183–1188. 67

[ZDL00] Zeng W., Daly S., Lei S.: Visual optimization tools in JPEG 2000. In *IEEE International Conference on Image Processing* (2000). 26, 37, 43, 74