Real-time Realistic Rendering And High Dynamic Range Image Display And Compression

Ruifeng Xu
University of Central Florida

REAL-TIME REALISTIC RENDERING AND HIGH DYNAMIC RANGE IMAGE DISPLAY AND COMPRESSION

by

RUIFENG XU

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the School of Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Fall Term
2005

Major Professor: Sumanta N. Pattanaik

ABSTRACT

This dissertation focuses on the many issues that arise from the visual rendering problem. Of primary consideration is light transport simulation, which is known to be computationally expensive. Monte Carlo methods represent a simple and general class of algorithms often used for light transport computation. Unfortunately, the images resulting from Monte Carlo approaches generally suffer from visually unacceptable noise artifacts. The result of any light transport simulation is, by its very nature, an image of high dynamic range (HDR). This leads to the issues of the display of such images on conventional low dynamic range devices and the development of data compression algorithms to store and recover the corresponding large amounts of detail found in HDR images. This dissertation presents our contributions relevant to these issues.

Our contributions to high dynamic range image processing include tone mapping and data compression algorithms. This research proposes and shows the efficacy of a novel level set based tone mapping method that preserves visual details in the display of high dynamic range images on low dynamic range display devices. The level set method is used to extract the high frequency information from HDR images. The details are then added to the range compressed low frequency information to reconstruct a visually accurate low dynamic range version of the image.

Additional challenges associated with high dynamic range images include the requirements to reduce excessively large amounts of storage and transmission time. To alleviate these problems, this research presents two methods for efficient high dynamic range image data compression. One is based on the classical JPEG compression. It first converts the raw image into RGBE representation, and then sends the color base and common exponent to classical discrete cosine transform based compression and lossless compression, respectively. The other is based on the wavelet transformation. It first transforms the raw image data into the logarithmic domain, then quantizes the logarithmic data into the integer domain, and finally applies the wavelet based JPEG2000 encoder for entropy compression and bit stream truncation to meet the desired bit rate requirement. We believe that these and similar such contributions will make a wide application of high dynamic range images possible.

The contributions to light transport simulation include Monte Carlo noise reduction, dynamic object rendering and complex scene rendering. Monte Carlo noise is an inescapable artifact in synthetic images rendered using stochastic algorithm. This dissertation proposes two noise reduction algorithms to obtain high quality synthetic images. The first one models the distribution of noise in the wavelet domain using a Laplacian function, and then suppresses the noise using a Bayesian method. The other extends the bilateral filtering method to reduce all types of Monte Carlo noise in a unified way. All our methods reduce Monte Carlo noise effectively.

Rendering of dynamic objects adds more dimension to the expensive light transport simulation issue. This dissertation presents a pre-computation based method. It pre-computes the surface radiance for each basis lighting and animation key frame, and then renders the objects by synthesizing the pre-computed data in real-time.

Realistic rendering of complex scenes is computationally expensive. This research proposes a novel 3D space subdivision method, which leads to a new rendering framework. The light is first distributed to each local region to form local light fields, which are then used to illuminate the local scenes. The method allows us to render complex scenes at interactive frame rates.

Rendering has important applications in mixed reality. Consistent lighting and shadows between real scenes and virtual scenes are important features of visual integration. The dissertation proposes to render the virtual objects by irradiance rendering using live captured environmental lighting. This research also introduces a virtual shadow generation method that computes shadows cast by virtual objects to the real background.

We finally conclude the dissertation by discussing a number of future directions for rendering research, and presenting our proposed approaches.

Dedicated to my grandparents

ACKNOWLEDGMENTS

There are many people who deserve my huge amounts of thanks and credit for their help along the way. First and foremost, I express my sincere thanks to my advisor, Dr. Sumanta Pattanaik, for his unhesitating support and steadfast guidance through my whole PhD study period in the past four years. He was always available when I had difficulties during my research. Without his encouragement and inspiration, this dissertation would have never been possible.

I devote my special thanks to Dr. Charles Hughes, my internal research committee member, for his very kind support and help in preparing this dissertation. His encouragement and confidence were of great help to build up my research interest and direction.

My sincere thanks also go to Dr. Erik Reinhard, my internal research committee member, and Dr. David Kaup, my external research committee member. They generously offered me their time and expertise in helping me refine my dissertation to meet the requirements. The much advice from them is indispensable to complete my dissertation.

To my colleagues Yugang Min, Yinghua Hu, Xin Bai, Danzhou Liu, Weifeng Sun, for the friendship from them. Also to the departmental staff, Jenny Shen, Cherry Tran, Linda Locky, Julie Faris, Nancy Barrett, Denise Tjong, Donald Harper, Robert Traub, for their various helps. The list of people who deserve my thanks is too long to completely write down all of them here.

Lastly and certainly, my parents and my sisters deserve my gratitude. They always back me in my many years of study. I am indebted to them for their constant love, encouragement and understanding.

ATI Corporation provided financial sponsorship to this research.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

term	*definition*
Bayesian method	A statistical inference technique used for estimating the conditional probability of an event A given event B, denoted as P(A|B), from the conditional probability of B given event A, denoted as P(B|A), and a prior probabilities of A and B, denoted as P(A) and P(B) using formula P(A|B)=P(B|A)P(A)/P(B).
bilateral filtering	A simple, non-iterative technique for edge-preserving smoothing. The filter kernel combines a range function and a distance function.
bit rate	In the image compression field, this term specifically refers to a measure to describe the compression rate of image encoding methods. It is computed as the average number of bits of each pixel in a compressed image.
bounding box	A virtual box that tightly encloses a scene.
BRDF/BTDF	Acronym for Bidirectional Reflectance/Transmittance Distribution Function. It is a four dimensional function that describes the radiance reflected/transmitted along outgoing directions as a function irradiance incident from any incoming direction.
bump mapping	A texture mapping technique that maps an image of vertex normals to a surface.
CABAC	Context-based Adaptive Binary Arithmetic Coding. An arithmetic coding method employed in H.264/MPEG-4 Part 10.
caustic	A bright pattern formed on a diffuse surface due to aggregation of lights.

codec	Acronym for COder/DECoder. Algorithms to encode and decode data.
PCA	Acronym for Principal Component Analysis. A mathematical method that uses a few basis vectors to denote a large collection of vectors.
CSF	Acronym for Contrast Sensitivity Function. A function that describing human eyes' just noticeable contrast under different signal frequencies.
DCT	Acronym for Discrete Cosine Transform. It is widely used image transformation method in image compression transform pixel blocks into coefficient blocks in terms of cosine functions of integer frequencies.
direct component	The lighting contributions directly from the light sources.
DirectX	A multimedia package by Microsoft Corporation.
displacement mapping	A texture mapping technique that maps an image of vertex displacements to a surface.
environment mapping	A technique that maps an environment to the surface of an object to create the visual simulation of an object illuminated by the environment.
exitant radiance	Radiance leaving a point.
GI	Acronym for Global Illumination. A rendering method that accounts for all features of light transports, e.g., inter-reflection.

GPU	Acronym for Graphics Processing Unit. It is a programmable SIMD processor, specifically designed to carry out many graphics computation fast and in parallel pipes.. At the current time the processing power of GPUs are increasing at a much faster rate than that of the standard CPUs.
HDR	Acronym for High Dynamic Range. In this thesis HDR corresponds to the dynamic range of pixel intensities of images of accurately simulated or captured lighting of the 3D world. Dynamic Range is the ratio between the highest pixel intensity and nonzero lowest pixel intensity. A range of 3 or more orders of magnitude is called high dynamic range.
hierarchical methods	A class of methods that first solve problems in global scale, then solve the sub-problems in local scale.
hit test	A process to find the nearest geometric element along a ray.
HMD	Acronym for Head Mounted Display. A head worn device that shows the image on its pair of tiny displays placed in positions such that they appear in front of the wearer’s eye. It is widely used in virtual reality and/or mixed reality applications.
HVS	Acronym for Human Visual System. It includes human eye, retina and associated circuits, visual cortex and any other parts of the human brain that deal with visual processing. HVS is responsible for receiving the external light stimulus, transforming it to the neural signal, and finally processing it to create appropriate visual perception.

image detail	High frequency information of an image.
image space	The two dimensional space to which images belongs.
importance sampling	The sampling process that generates random samples whose distribution follows some probability density function.
incident radiance	Radiance reaching a point.
indirect component	A synthetic image generated by taking account of illumination due to indirect light sources.
irradiance	A radiometric unit of light measurement. It is the flux incident on unit surface area.
irradiance gradient	The gradient of irradiance with respect to either translation or rotation.
JPEG	A popular DCT based lossy image compression standard. It originated as Acronym for Joint Photographic Exerts Group
JPEG2000	A wavelet based image compression standard. It was proposed by the Joint Photographic Exerts Group in 2000. Original JPEG standard was proposed in 1992.
JPEG-LS	A DCT based lossless image compression standard.
kd-tree	A data structure that accelerates the searching operation within a collection a multi-dimensional data.
Laplacian function	A bell-shaped function whose shape is controlled by two parameters.

LCIS	An acronym for Low Curvature Image Simplifiers. It is an edge preserving smoothing method used in a HDR tone-mapping algorithm proposed by Jack Tumblin and Greg Turk in 1999.
LDR	Acronym for Low Dynamic Range. The conventional display devices have a typical dynamic range of 2 orders of magnitude, which is often called LDR to distinguish from HDR associated with natural images and rendered images.
Level Set methods	A class of numerical algorithms for simulation of the movement of dynamic implicit surfaces and approximation of solutions to the Hamilton-Jacobi partial differential equation.
light field	The collection of radiance on any point in the scene along any direction.
LOD	Acronym for Level Of Detail. In the context of this thesis LOD method is used for choosing optimal number of polygons for describing a scene geometry from particular view such that the visual quality is not degraded.
lossless compression	The data compression technique to compress data in such a way that the original data can be exactly recovered.
lossy compression	A data compression technique that is not lossless. Normally used in image compression and produces images that aggressively reduces the image data size.

Metropolis algorithm	A rejection based Monte Carlo sampling technique for sampling any probability distribution. The algorithm generates a sequence of samples from the joint distribution of two or more variables.
MR	Acronym for Mixed Reality. A research field that integrates virtual scenes and real scenes.
MJPEG	Acronym for Motion JPEG. A video compression standard.
Monte Carlo methods	Techniques for estimating the solution of a numerical or mathematical problem by means of random sampling experiments.
Monte Carlo noise	An artifact in results obtained using Monte Carlo methods due to insufficient sampling .
MSE	An acronym for Mean Squared Error. An objective error measurement computed as the mean of the squared errors.
natural image	A real world image.
octree	A data structure to accelerate searching operation in 3D space. It is a tree data structure with internal nodes having up to eight children.
OpenGL	A popular graphics programming interface.
parameter space	A space to which the parameters belong.
Photometry	The measurement of the light taking into consideration the effect of light on HVS.

Photon mapping	A global illumination algorithm based on ray tracing used to realistically simulate the interaction of light by tracing photons from light source(s).
image profile	The low frequency information of an image.
PRT	Acronym for Pre-computed Radiance Transfer. A technique to pre-compute part of the light transport and use it for real-time global illumination.
PSNR	Acronym for Peak Signal-to-Noise Ratio. The ratio between the maximum value of a signal and the magnitude of background noise.
QC	Acronym for Quantization Coefficient. It is the quantization step used in JPEG.
radiance	A light measuring unit in radiometry. It is the flux per projected unit area per solid angle.
radiometry	The measurement of radiant electromagnetic energy.
radiosity	A light measuring unit in radiometry. It is the flux leaving unit area. Radiosity is also used to represent the finite element algorithm used to compute the radiosity distribution in a scene.
reflectance	Ratio between reflected light (mostly radiance) and incident light (mostly irradiance).
reflection	A light transport phenomenon by which light bounces from the incident surfaces.

refraction	A light transport phenomenon by which lights change direction when entering from one medium to another medium with different refractive index.
range compression	An operation to compress the high dynamic range of an image to display it on a LDR display device.
ray path	The path that a ray follows in space.
ray tracing	An algorithm to render a scene by tracing rays along its reflected and refracted directions.
rendering	The process of creating synthetics images. It includes the simulation of light in three-dimensional scene; generating the actual image by projection.
Rendering Equation	A Fredholm integral equation of second kind that describes the relationship between outgoing light and incoming light.
RLE	Acronym for Run Length Encoding. A data compression algorithm that replaces consecutive repeated data values with the value and its run.
RMS	Acronym for Root Mean Square. It is the square root of the arithmetic mean of the squared set of values.
rotational gradient	Gradient with respect to rotational changes.
sampling rate	The rate at which an analog signal is sampled for conversion to and from the digital domain.
soft shadow	Shadow with blurry/smooth boundaries.

spatial coherence	The similarity between spatially neighboring elements.
SH	Acronym for Spherical Harmonics. A group of spherical basis functions.
shadow test	A process to find if the light source is occluded.
sub-band	A component that captures the information content of an image within some frequency ranges.
subsurface diffusion	A light transport phenomenon in which light scatters randomly in translucent materials.
synthetic image	An artificial image generated using a computational algorithm.
temporal coherence	The similarity between temporally neighboring elements.
tone mapping	An operation to map HDR images to LDR images.
translational gradient	The gradient with respect translational changes.
transmittance	The ratio between refracted light and incident light.
VDP	Acronym for Visual Difference Predictor. A numerical computation technique that computes subjective visual difference between two images.
volumetric scattering	A light transport phenomenon in which light scatters along all directions by interacting with every particle of the medium such as cloud, smoke, fog.
wavelet	A basis set of mathematical functions that are only non-zero within a limited spatial domain.

world space	The universal 3D space.
YC_bC_r	A color space mainly used in image and video compression.
Z-buffer	A portion of the local memory that is used to store depth information of points visible through each pixel in an image frame.

CHAPTER ONE: INTRODUCTION

The research presented in this thesis focuses on realistic rendering of synthetic images, a process that involves the sequence of computation steps outlined in Figure 1.1. Given the scene geometry and associated material property, the rendering process starts with the simulation of light transport in the scene. Light originating at the light source gets distributed in the scene through a complex series of interaction of light and matter. Simulation of this process is commonly referred to as global illumination computation. Accurate computation of global illumination is key to realism in synthetic images. Even after 25 years of research on this computation problem, developing algorithms for efficient and accurate lighting computation is still an active area of research. The step following the lighting simulation is the actual rendering of the 2D image for a given virtual camera definition. This rendering is carried out by using a Z-buffer or ray-tracing based technique. Though described here as two distinct steps, it is possible to combine the global illumination computation and image rendering into a single step. Rendering based on such an approach is called image-space rendering and the other approach is called an object-space technique.

The light intensity captured on the pixels of an image computed from a globally illuminated scene can have dynamic range higher than the range available on most conventional display devices. For accurate display of these high dynamic range (HDR) images, the pixel intensity range must be compressed to match the display device in such a way that the perceived appearance of the display image is representative of the actual appearance of the rendered scene, if it existed. Such range compression processes are called tone mapping. Tone mapping is an active area of research. Because of the high dynamic range of intensities among the pixels, the representation of HDR images requires more than 8 bits per pixel per color channel. There does not exist any standard data compression technique for compressing such images. Although the HDR image data compression is not a critical component of the rendering process, because of the problems involved in storage and transmission of HDR image data, their compression issue has become a relevant topic to realistic rendering. The research presented here covers almost all these steps of rendering from lighting computation to image display.

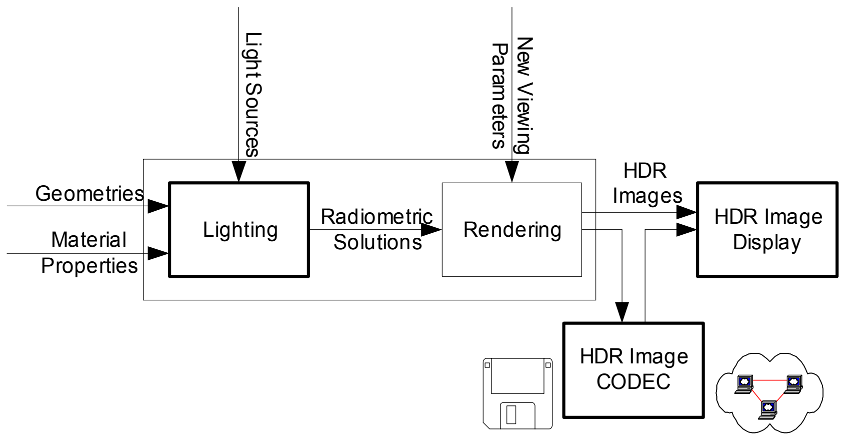


Figure 1.1: The Framework of Realistic Rendering

Issues involved in rendering are many. The following discussion focuses on the issues most relevant to this study. These are: Real-time realistic rendering, efficient lighting computation, tone mapping and HDR data compression.

Dynamic range of illumination in real world scenes is often more than four orders of magnitude (from highlight to shadow). The synthetic images generated from accurate simulation of the lighting phenomena in the real world can thus produce images with high dynamic range. Such image pixels must be encoded in more than one byte per color channel. This leads to difficulties not only with display but also with storage and transmission. A tone mapping operation is necessary to compress the high dynamic range of the images so that they can be displayed on conventional low dynamic range display devices, while retaining most of the perceptual cues associated with the image. Data compression is necessary in order to alleviate the heavy burden on the storage and transmission due to high data volume in raw image formats.

1.1 Real-Time Rendering

Real-time lighting simulation remains a hard problem. The rendering equation [Kajiya 1986] describes the relationship between the outgoing radiance and the incoming radiance. Synthetic images generated from a complete solution of this equation are visually indistinguishable from real photographs. Though it is now possible to compute such images, the time required to generate a high quality image conflicts with its use in interactive environments. Radiosity [Greenberg 1986], Monte Carlo ray tracing [Kajiya 1986] and photon mapping [Jensen 1996] are three major algorithms proposed to numerically solve rendering equation. However, the amount of processing power required for straight forward use of each of these algorithms is too high for today's personal computers and hence precludes real-time computation, which is desired in many practical applications, like games and military training.

Inspired by *irradiance volume* [Greger 1998; Nijasure 2003], a novel 3D space subdivision is proposed in this research to accelerate global illumination for real-time performance. The space enclosing the scene is first subdivided until each local scene is small enough so that its local lighting condition can be approximated as distant environment lighting. During the rendering, the sources distribute their light to the local scenes, and each local scene is then rendered using its local lighting condition.

Global illumination of static scenes remains an expensive computation problem. Adding dynamics (character animation) makes global illumination even more challenging. We address only a specific aspect of real-time rendering of dynamic scenes, this is, real-time rendering of environment lighting in dynamic scenes. Pre-computed radiance transfer (PRT) [Sloan 2002] is a new approach to real-time environment lighting. With the incident radiance to each vertex pre-computed, real-time rendering is achieved by performing a few spherical harmonics coefficient multiplication and addition operations for each vertex [Ramamoorthi 2001]. The use of PRT makes the solution of this previously daunting global illumination problem possible in real-time. A straightforward extension to dynamic objects is to pre-compute each animation frame for use in the later rendering stage, but this will produce a huge volume of data. This quantity of data will prevent real-time rendering performance by requiring substantial time to load data from disk to memory. The approach presented in this research addresses this problem. The object surface is unfolded to a 2D parameter plane, where each point is associated with the PRT of its corresponding 3D point on the object surface. The "image" of the PRT is then compressed. The compression drastically reduces the data volume and thus allows the rendering task to be carried out in real-time. Moreover, the choice of sampling rates and compression algorithms can lead to a variety of level-of-detail strategies, supporting the applicability of this approach to complex scenes.

The increase in computation power available in today's programmable graphics hardware (a.k.a. GPU) promises real-time rendering solution of rendering equations. However, the Single Instruction Multiple Data (SIMD) execution model of these GPUs requires that considerable amount of efforts are necessary to develop new GPU lighting simulation algorithms or to port well-known CPU algorithms into GPUs. Despite the fact that over the years great progress has been made along this direction real-time realistic lighting and rendering will remain a research focus for many years to come.

1.2 Efficient Monte Carlo Based Global Illumination Computation

Monte Carlo methods [Kajiya 1986; Lafortune 1993; Ward 1998] for global illumination computation compute the radiance value for each pixel using random sampling techniques. These methods are more general in that they can handle a variety of surface geometry and a variety of surface properties. However, they have their own drawback, in that Monte Carlo rendered images tend to be noisy. According to the noise analysis in [McCool 1999], plenty of samples are needed for each pixel around illumination discontinuities such as light source edges, penumbrae, fuzzy specula reflections, and caustics to obtain estimates under some threshold error [Rushmeier 1994; Purgathofer 1987]. This significantly impedes the rendering speed for a high quality synthetic image. As a result, noise free computation of reasonably complex images may takes minutes to hours [Ward 1998; Shirley 1996], which is generally not affordable in practice. So, when using a Monte Carlo based global illumination method for image synthesis, one trades rendering time for noisy images. The amount of noise in the image varies as an inverse square relationship to the rendering time. It is impractical trying to get high-quality synthetic images simply by increasing sampling size [Rushmeier 1994]. In other words, images computed within a reasonable time period using a Monte Carlo radiance computation method will invariably have some noise.

To remove these noises, research has devised various methods, which can generally be categorized into two classes: post-processing and importance sampling. The former removes the Monte Carlo noise after rendering, while the latter suppresses the Monte Carlo noise during rendering. This research proposes two post-processing approaches for noise reduction. Using these two approaches it is possible to create visually pleasing synthetic images without increasing the number of samplings.

Three keen observations inspire this research on Monte Carlo noise reduction. First, diffuse inter-reflection is responsible for most of the noise [Jensen 1995]. Second, diffuse inter-reflection tends to be of low frequency [Ward 1988]. Third, most of the noise concentrates around high frequency illumination regions [Rushmeier 1994]. In one of our methods we exploit the idea of Monte Carlo noise reduction using Bayesian method. Bayesian denoising is a successful image denoising technique that opportunistically suppresses the high frequency information where noise concentrates. This approach builds a statistical model of Monte Carlo noise, and removes the noises using a Bayesian method based on this model. Our second method is based on the fact that Monte Carlo noise appears both as outliers and as inter-pixel incoherence in a typical image rendered at low sampling density [Jensen 1995; Lee 1990; McCool 1999; Rushmeier 1994]. Unfortunately, none of previous approaches can reduce both types of noise in a unified way. This drawback also inspires us to propose a unified Monte Carlo noise reduction approach to suppress both outliers and inter-pixel incoherence using bilateral filtering [Tomasi 1998].

1.3 High Dynamic Range Image Tone Mapping

High dynamic range (HDR) images are a natural outcome of various renders [Ward 1997]. Using a method proposed by computer graphics researchers [Debevec 1997, Mitsunaga 1999], it is now possible to capture high dynamic range real world images from a series of photographs with different exposures. Thus HDR images have become more and more popular and important in computer graphics research and applications [Kollig 2003; Cohen 2002; Debevec 1998]. However, HDR also brings challenges for image display, because the existing display devices, including CRT monitors, printers, liquid crystal displays, are low dynamic range devices. Simple scaling based mapping of image pixel intensities to display pixel causes a significant loss of visible details. So, advanced range compression techniques are necessary to display HDR images in a visually realistic way on standard display devices [Durand 2002; Fattal 2002; Reinhard 2002; DiCarlo 2001; Pattanaik 1998].

The problem of high dynamic range image compression has been formulated [Tumblin 1999a,b] as a non-linear range compression problem satisfying the following conditions: separate the image into a profile image and a detail image, leave the details unchanged; compress the profile image into a lower dynamic range. The profile contains low frequency information and the high dynamic range. The detail contains high frequency information. The first stage of this approach is to separate the profile and the detail from the signal. In this research we propose a level set based approach to carry out this separation. We then use any available range compression technique to compress the dynamic range in the profile image and then combine the profile and detail images to reconstruct a low dynamic range version of the original image.

1.4 High Dynamic Range Image and Video Data Compression

The raw HDR image/video data size is daunting. The raw three-float representation is 4 times the size of a traditional RGB image/video. The RGBE (or XYZE) encoding scheme [Larson 1991] reduces the size from three floats per pixel to four bytes per pixel and thus makes the image only 1.33 times larger. Other encoding schemes such as Pixar's Log, SGI's LogLuv [SGI 1997], or ILM's OpenEXR [ILM 2004] have also been proposed to reduce the size while preserving the dynamic range. However, no standard compression scheme is currently available to facilitate common use of HDR images/videos like standard LDR counterparts [Pannebaker 1993; Pannebaker 1995, Salomon 2000]. This research is inspired by the compression technique used in standard image/video codecs that makes use of spatial and temporal coherence between neighboring pixels. We build our HDR image and video codec from an existing image and video codec. In addition to making the development simple and efficient, the use of the mature techniques of conventional image and video codecs allows their streaming capability to be directly applied to stream HDR video over networks.

One of our two proposed methods uses RGBE format as the starting HDR image format. In the RGBE format each pixel is encoded as a three byte color base and one byte common exponent. We compress the color base using standard lossy or lossless image and video compression techniques, but compress the common exponent using only a lossless approach. The differential treatment between common exponent and color base is due to the fact that the image quality is very sensitive to error introduced by the common exponent. This method can be adapted to use other similar HDR pixel formats, like XYZE [Larson 1991] and LogLuv [Larson 1998].

Our other compression method extends JPEG2000 to compress HDR images. In this approach we exploit the fact that JPEG2000 supports up to 16 bits integer. We log encode the floating point values in each channel and follow it up with an adaptive quantization technique to convert each color channel into 16 bits. The resulting lossy compression preserves the image quality even at very high compression rates.

1.5 Contributions of This Research

We list here our research contributions to the field of realistic rendering.

This study presents a novel hierarchical rendering algorithm that combines 3D space subdivision and pre-computation for global illumination computation of complex scenes in real-time.

Dynamic objects change their position or shape with time so that part or whole of the acceleration structure and cached values used in rendering algorithms become invalid between frames. This research proposes a novel pre-computation based method to render dynamic objects in real-time.

Effective Monte Carlo noise reduction is a better alternative to simply increasing the sampling rate to generate images of the same quality image. A statistical Monte Carlo noise model in wavelet domain is proposed and a Bayesian based approach is applied to reduce Monte Carlo noise. This research also proposes bilateral filtering in Monte Carlo noise reduction.

A novel level set based method for tone mapping of HDR image is proposed to allow their display on LDR display devices without loss of details.

To reduce the high data volume of raw HDR images, this research explores the applications of DCT and wavelet based methods of HDR image compression.

1.6 Organization of This Dissertation

This thesis is organized into three parts and 9 chapters. This is the first chapter. The second chapter gives a literature review, and the last chapter gives conclusion and discusses some future work. The other six chapters represent research contributions. They comprise the following three main parts of the thesis.

NEW HDR IMAGE DISPLAY AND DATA COMPRESSION METHODS [Chapter 3, 4]

A level set based HDR tone mapping method is first presented. The method is based on the idea of separating HDR image into a profile image and a detail image. The profile image is range compressed to reduce the dynamic range and the detail image is added back to the compressed profile to reconstruct a low dynamic range image.

Two HDR image and video data compression methods are then described. The methods are extensions to the existing discrete cosine transform (DCT) based and wavelet based JPEG compression standards. These two methods support both lossless and lossy HDR data compression.

NEW MONTE CARLO NOISE REDUCTION ALGORITHMS [Chapter 5]

Two post-processing based methods are presented for noise reduction in synthetic images created using Monte Carlo based rendering algorithms. One of the methods models the noise distribution in wavelet domain and reduces it from the image using a Bayesian approach, and the other uses a modified bilateral filtering method for noise removal. The Monte Carlo noise reduction allows high quality image to be rendered with less samples, thus saves rendering time.

NEW FAST RENDERING APPROACHES [Chapter 6, 7, 8]

A 3D space subdivision method is proposed, which is used in pre-computed cache based rendering approach to enable real-time global illumination.

A pre-computation based method is then described that renders dynamic objects at interactive rates.

Finally, several rendering methods are proposed to solve the rendering issues to satisfy the “consistent lighting on virtual and real objects” requirement in mixed reality.

CHAPTER TWO: BACKGROUND

Rendering is an important area of computer graphics dealing with computation of realistic images of three dimensional scenes. Its potential for wide applications has drawn the attention of a number of computer graphics researchers. Over 30 years' research in this field has led to its popular application in today's entertainment, industrial design, and training simulation, etc. As it is impractical to review all this work in this thesis, we will only present the background research that is most relevant to the research reported here. Basic knowledge of light simulation is first described briefly, followed by a summary of previous research efforts in Monte Carlo noise reduction techniques and light transport simulation acceleration algorithms, and a discussion of previous HDR image processing methods.

2.1 A Brief Account of Lighting Simulation

The core of the rendering problem is light transport simulation [Greenberg 1997]. The inputs to the light transport simulation are the material properties, the light source(s), and the scene geometry. The output is either light energy distribution or an image whose pixel values are either radiometric or photometric quantities.

Rendering time is mostly dominated by the light transport simulation. Various factors make this solution expensive. They are the complexity of light sources, the complexity of material properties, and the complexity of the light transport.

The material properties specify the type of interaction between the material and the light. The main interactions of interest are reflection and transmission, although volumetric scattering and subsurface diffusion are also considered. Reflectance and transmittance are often represented as simple mathematical models. Accurate BRDF and BTDF functions are slowly getting wide acceptance. The Phong model [Phong 1975] and Blinn model [Blinn 1977] are the simplest of the models for the diffuse and specular reflections and refractions of light, and are popularly used in the computer graphics community. The Cook-Torrance model [Cook 1981] is based on micro-facet for more realistic approximation. The Ward model [Ward 1992] and Lafortune model [Lafortune 1997] are relatively simple mathematical models and are used to fit measured data and the resulting BRDFs often provide accurate representation of the surface reflectance properties.

There are various types of light sources with different shapes and light distribution. Point light sources and directional light sources are most popularly used in rendering, but linear light sources, area light sources, and volumetric light sources are more common in the real world.

Light transport simulates the propagation of light that originates from light sources and reflects (refracts or scatters) inside the scene. The equilibrium distribution of light for scenes with reflective materials is concisely described by an integral equation [Kajiya 1986, Pattanaik 1993] of the second kind. It is known as the rendering equation and is shown in Equation (2.1).

$$L(w_o) = L_e(w_o) + \int_\Omega L_i(w_i) f_r(w_i, w_o) \cos\theta_i dw_i \tag{2.1}$$

where, θ_i, w_i, w_o are the incident pitch angle, the incident direction, and the exitant direction, respectively; L_i, L are the incident radiance and the equilibrium exitant radiance; L_e is the radiance due to self emission; and f_r is the BRDF.

A solution to the rendering equation is also known as *Global Illumination*. An analytical solution to the rendering equation is possible in only very special and simple cases. Thus a numerical solution is indispensable in almost all cases.

It is convenient to rewrite the rendering equation, Equation 2.1, in terms of light transport operator as Equation (2.2) to explore various solution strategies.

$$L = L_e + TL \tag{2.2}$$

As in the earlier equation the light distribution over scene surfaces in its initial and equilibrium state are denoted as L_e and L, and the light transport operator that is responsible for one bounce of light is denoted as T.

Iterative substitution of T on the right side of Equation (2.2) by L_e+TL, the rendering equation can be represented in expansion form, as shown in Equation (2.3), which can be implemented using a recursive ray tracing algorithm.

$$L = L_e + TL_e + T^2 L_e + \cdots \tag{2.3}$$

It is also possible to rewrite Equation (2.2) in recursive form, as shown in Equation (2.4).

$$\begin{aligned} L_0 &= L_e \\ L_n &= L_e + TL_{n-1} \\ \lim_{n \to \infty} L_n &= L \end{aligned} \tag{2.4}$$

This formulation leads to a fast solution by iterating light bounces recursively. We will refer to these formulations at subsequent chapters dealing with rendering of the complex scenes.

There are a few other ways to formulate the light transport phenomenon, like GRDF [Lafortune 1994], particle transport [Arvo 1990], photon scattering [Jensen 1996], and importance transport [Pattanaik 1993], which could lead to new solution strategies.

Among various rendering algorithms proposed so far, some of them are worth discussion because they are either widely used or heavily researched. Z-buffer based graphics pipeline is the only widely adopted rendering algorithm in the industry because it can render scenes in real-time.

Radiosity, Monte Carlo ray tracing and Photon mapping can provide GI features that lack in Z-buffer based graphics pipeline.

STANDARD GRAPHICS PIPELINE [Catmull 1974] is one of the earliest proposed and still widely used rendering algorithms. It combined the Z-buffer based image computation technique with simple direct lighting and ambient lighting to simulate light transport. Its simple features made it amenable to direct implementation in hardware. Ever increasing speed of the graphics hardware gave way to many innovations in the graphics pipeline and ultimately [Purcell 2002; Purcell 2003] gave way to the programmable graphics pipeline. This programmable feature enables implementations of many complex solutions to the rendering equation (discussed below) on fast graphics hardware. Recent algorithms such as environment mapping and pre-computed radiance transfer (PRT) were designed to take advantages of graphics hardware to render complex illumination features in real-time.

RADIOSITY [Goral 1984] is the first global illumination algorithm. It breaks the scene into many small elementary patches, and computes the form factors between any pair of patches. The final rendering result is obtained by solving a linear system. It provides an accurate solution to light transport simulation of diffuse scenes, but it has difficulty in handling non-diffuse surfaces and shadow boundaries. Difficulties in appropriate scene discretization and time-consuming accurate computation of form factors make this method, less practical as an accurate global illumination solution technique.

MONTE CARLO RAY TRACING [Kajiya 1986] is an important numerical algorithm to solve the rendering equation. It extends Whitted's recursive ray tracing [1980] by employing Monte Carlo methods to sample rays. It is powerful enough to provide all global illumination features, like diffuse inter-reflection and caustics, but it is notoriously very slow, which restricts its wide application. It is not uncommon for a ray tracer to take a few hours to render a moderately complex scene in decent quality. Besides the bounding box and the octree, caching techniques are also exploited as efficient acceleration techniques. Monte Carlo noise is an inescapable artifact in synthetic images rendered using Monte Carlo methods due to insufficient sampling rate.

PHOTON MAPPING [Jensen 1996; Pattanaik 1992] simulates light transport by distributing photons over the scene and computing their density. The photons are tiny energy packets, which are first distributed to the scene and stored in an auxiliary data structure in the distribution step. The gathering step computes the brightness by estimating the density of nearby photons. It is an accurate and efficient algorithm to compute some global illumination features like caustics.

A huge amount of research effort has been devoted to improving the above-mentioned algorithms. Hybrid algorithms have been devised to combine and benefit from multiple individual rendering techniques in a single rendering algorithm. Wallace et al's two-pass algorithm [Wallace 1987] combines benefits of *radiosity* and *ray tracing* by first computing a view-independent radiosity solution and then synthesizing a view-dependent ray tracing image. Chen's multi-pass rendering [1991] combines radiosity, Monte Carlo path tracing and light ray tracing in a progressive rendering framework. Keller's instant radiosity [Keller 1997] simulates global illumination by combining multiple graphics pipeline rendering results. And Ward's *Radiance* software [1994] computes specular reflections in a deterministic manner, and diffuse inter-reflection in a stochastic algorithm. Lafortune et al's bi-directional path tracing [1993] traces the rays from both the light sources and the eyes. Veach's Metropolis light transport [1997] algorithm employs the Metropolis's sampling technique to sample the ray path space. Other important rendering acceleration techniques consist of Level of Detail (LOD) [Funkhouser 1993], scene simplification [Luebke 1997], and perceptual information acceleration [Bolin 1998].

Realism and *real-time* are the ultimate goals of rendering research. The realism spectrum of rendering algorithms spans direct lighting with ambient through full global illumination solution. High realism requires not only multiple bounce light transport simulation, but accurate BRDFs and light source emission. The acquisition of accurate geometries, BRDFs and light sources are also important components of a rendering solution, but their discussion is beyond the scope of this thesis, which devotes its efforts towards Monte Carlo noise removal, real-time light transport simulation, visual display and data compression of high dynamic range synthetic images.

2.2 Monte Carlo Methods and Acceleration Techniques

The Monte Carlo method [Rubinstein 1981] is a numerical simulation method to solve problems using a collection of samples from some distribution. It is a powerful tool to numerically approximate an integral with high dimensional kernels, which makes it a natural choice to compute the rendering equation since it has a kernel of infinite dimension. Because of their versatility, Monte Carlo methods dominate current research in global illumination. Like any numerical solution method, Monte Carlo methods introduce error into the solution. In synthetic images generated using Monte Carlo methods such errors appear as noise artifacts. In the remainder of this thesis, we will refer to this noise as Monte Carlo noise.

2.2.1 Monte Carlo Noise Reduction

Monte Carlo noise reduction is research directed towards synthesizing high quality images. Increasing the number of samples reduces errors, but is less efficient. Rushmeier and Ward [Rushmeier 1994] provide a time measure in terms of the number of samples required to get a given accuracy. The expression for this time measure is given in Equation (2.5)

$$M_t=(4S_{p\_16} / \Delta L_{tvis})^2 \qquad (2.5)$$

where, M_t denotes the estimated number of samples required to obtain a result with accuracy or error tolerance of ΔL_{tvis}, S_{p_16} is the mean error obtained using 16 samples.

Thus, to double the accuracy, i.e. to reduce ΔL_{tvis} to one half of its original value, M_t should be increased four times. If we assume that we are using a Monte Carlo path-tracer to compute radiance using path lengths of size 5 on average in order to reduce the error by half, we have to increase the computation time by $4^5=1,024$ times. Similarly, in order to increase the image quality four times, we have to increase the computation time by $16^5=10^6$ times.

Post processing based noise reduction has shown promise as an efficient Monte Carlo noise reduction technique. Although any classic image noise reduction algorithm can be used to reduce Monte Carlo noise, a naive application is not always very effective. So, much work has been devoted to carry out effective and efficient post-processing of Monte Carlo noise [Lee 1990; Rushmeier 1994; Jensen 1995; Tamstorf 1997; McCool 1999]. A comprehensive account of this literature is given in [McCool 1999].

The work of [Lee 1990] revealed that Monte Carlo noise (for sensory data) comes from frequencies above the sampling frequency being folded into frequencies that can be displayed, and presents itself as noisy spikes. Lee proposed median and alpha-trimmed mean filters to reduce the noise, which assumes the noise comes from unwanted secondary input, such as bit errors in transmission, and thus just throws away those extraneous samples. However, such techniques are not very effective. It is mostly because the assumptions behind the use of the techniques are not valid for synthetic images. The noise in synthetic images comes from insufficient sampling, not secondary input, and the noise carries information about the true value of the image [Purgathofer 1987; Rushmeier 1994].

Rushmeier [1994] first realized that the Monte Carlo noises in a synthetic image carry meaningful information and proposed an energy preserving non-linear filter to make use of the information implicit in the noise by distributing them to neighboring pixels. Their energy preserving method [1994] suppressed the *error* by distributing the error of image pixels to nearby pixels. However, its use of an average of a number of samples to approximate the rendering equation is based on the assumption that a simple linear tone operator will be used for display of the resulting images. As we mentioned earlier, HDR image is a natural outcome of most accurate renders and the use a non-linear tone reduction operator to present the final image on existing devices is a common practice. This invalidates the assumption made by Rushmeier's method, and hence, invalidates the derivation of the formulation for the number of samples.

Jensen et al. [Jensen 1995] presented an alternative perspective to Monte Carlo noise in synthetic image. They assumed that the noise is mainly from diffusely reflected indirect illumination. They separated indirect illumination and removed the noise from this indirect component using various filters, such as low pass and mean filters. However, they failed to consider the "outliers" in areas with sharp luminance changes, like light source edges [Rushmeier 1994] and the filters used blurred the edges.

McCool proposed an anisotropic diffusion based method [1999] to reduce the noise by smoothing the pixels inside regions without blurring the edges and thus improved upon Jensen's approach. The denoising results look impressive. But, this approach is intended to solely remove the inter-pixel incoherence. It is also very slow because of its iterative nature.

The following summarizes the work done so far on Monte Carlo noise removal technique by listing various facts about Monte Carlo noises.

- Monte Carlo noise comes from insufficient sampling in high variance regions.
- The noise can be classified into two types: outliers [Rushmeier 1994] and inter-pixel incoherence [Jensen 1995; Tamstorf 1997].
- Outliers often occur where luminance changes abruptly, like edges.
- Outliers carry useful luminance information.
- Inter-pixel incoherence often occurs inside high coherent regions.
- Additional noise comes from computation of indirect diffuse inter-reflection.

Based on the above background behind Monte Carlo noise this thesis proposes two novel approaches to noise removal. One is based on bilateral filtering [Xu 2005a] and the other based on Bayesian theorem [Xu 2005b]. Detailed descriptions of these methods are given in Chapter 5.

2.2.2 Acceleration Techniques for Rendering

Realism and *rendering time* conflict with each other. Complex scenes and dynamic objects are common in real world applications, thus their real-time realistic rendering are worth research attention. The challenge in rendering complex scenes come from the involvement of a large amount of geometry in the visibility test, and additional challenges in rendering dynamic objects comes from the requirements to maintain acceleration data structures and caches.

The primary acceleration strategy in visibility computation (ray hit test, shadow ray test, etc.) is to use the *octree*, *bounding box* and *kd-tree* data structures to accelerate this step. The study presented in this thesis proposes a space subdivision technique to meet the challenge in realistic rendering of complex scenes. It is based on the observation that the light transport can be considered to happen in two scales: global light transport and local light transport. The whole scene can be spatially divided into local scenes. As long as the light field enclosing some local scene is known, it can then be rendered using the local light field as light sources. A new rendering framework is then possible by using the space subdivision technique: first distribute the light from the light sources to the local scenes; then render each local scene using the local light field. This study proposes a pre-computed light transport data compression technique for real-time rendering of dynamic objects. A number of additional strategies that are based on reducing the number of such tests are relevant to the work proposed in this thesis. They are caching, pre-computation, perception based rendering and rendering simplification. The following paragraphs provide a brief background to these techniques.

CACHING exploits coherence – similarity between neighboring elements in space and time. It is an important property to utilize to save computational time. The general idea to exploit coherence is to cache computed samples and to reuse them by interpolation or extrapolation instead of computing a new sample. There is extensive research in cache designs, which may happen in image space, world space, or line space. The cached data can be pixel intensity, irradiance, radiance, ray, etc. The intensity of neighboring pixels in an image tends to change smoothly, and this property is exploited in *render cache* [Walter 1999; Walter 2002; Bala 2003], in *tapestry* [Simmons 2000] and in *shading cache* [Tole 2002]. Neighboring rays also tend to have coherent radiance, and this property is equally exploited for efficient rendering [Gortler 1996; Levoy 1996; Bala 1999a; Bala 1999b; Bala 1999c; Teller 1996; Ward 1999; Dmitriev 2002; Drettakis 1997]. *Irradiance cache* [Ward 1988] is used in the *Radiance* software [Ward 1994] to solve the diffuse inter-reflection term of the rendering equation. It benefits from the coherence of irradiance values over the scene surfaces.

PRE-COMPUTATION based techniques pre-compute part of the light transport simulation and render the scene using the pre-computed partial light transport related data, such as Pre-computed visibility function [Heidrich 2000], Pre-computed Radiance Transfer (PRT) [Sloan 2002] Such pre-computation based techniques are mostly useful in computing far-field environment lightings. James and Fatahalian propose a real-time algorithm for deformable objects rendering by pre-computing their impulse response functions [2003].

PERCEPTUAL BASED RENDERING techniques are based on adaptively distributing more computational resources to areas of larger perceptual error and correspondingly less to other area. It can thus gain improvement in visual quality in the same computation time. Most notable is the work of Ramasubramanian et al. [1999] where the authors propose the computation of a physical error map derived from the limits of human visual system (HVS) to guide the global illumination computation. Unfortunately, the functioning of HVS is not well understood and this fact limits the development in the area of perception-based rendering.

RENDERING SIMPLIFICATION trades the computation efforts of less important accuracy for rendering time saving. The four dimensional BRDF function can be simplified into a low dimensional function that captures the dominant behaviors of the reflectance function while not perceptually hurting the realism of the final rendered results. For example, spherical harmonics can be used to convert a BRDF function to a coefficient vector thus transforming the light material interaction into either a vector dot product or a matrix vector product [Ramamoorthi 2001].

GPU PARALLELISM is the major mechanism employed by today's graphics hardware to gain high rendering performance. Prior to the introduction of the GPU, the main component of parallelism in rendering was to divide the image into patches, and to send these patches to multiple computers to render in parallel [Parker 1999, Wald 2001a; Wald 2001b; Wald 2002; Wald 2003]. The GPU has emerged to be a promising high performance platform for realistic rendering. Various rendering algorithms have been ported to the GPU, like ray tracing [Purcell 2002] and photon mapping [Purcell 2003]. Its superior performance as compared to CPU-based techniques derives from the fact that GPUs support data parallelism.

We have briefly described the major techniques to accelerate light transport simulation. Although they can significantly reduce the rendering time of complex scenes and dynamic objects, they still do not provide real-time rendering performance of complex scenes and dynamic objects. This study proposes a space subdivision based technique to render complex scenes efficiently [Xu 2005e], and a pre-computation based algorithm to render dynamic objects in real time [Xu 2004]. Our complex scene rendering algorithm shares some similarity with the work of Nijasure [2003], where he proposed to compute the incident radiance on uniform grid points using programmable graphics hardware. Our dynamic object rendering algorithm extends the PRT algorithms [Sloan 2002; Sloan 2003a; Sloan 2003b; Lehtinen 2003], which are limited to the rendering of static scenes.

2.3. High Dynamic Range Image Processing

As discussed in the previous chapter, the synthetic images resulting from accurate solution of light transport simulation are generally of high dynamic range (HDR). This HDR nature poses problems for their realistic display on conventional low dynamic range (LDR) presentation devices. This is the Tone Mapping (TM) problem. The HDR nature demands a larger number of data bits for their effective representation and consequently significantly increases the raw data size of the resulting images and videos. For example: a raw HDR image encoding the image data in 32 bit floating point representation per color channel is 4 times the size of a conventional 24 bits image of the same dimension. This increase in size poses problem in storage and transmission. Data compression is a solution to this problem. As in standard image/video compression, lossy compression is an effective way for providing aggressive data compression. In this section we provide some background on tone mapping and lossy image/video compression.

2.3.1 Tone Mapping

A tone mapping algorithm attempts to convert a HDR image to a LDR image in such a way that the perceived tones (such as brightness, colorfulness, contrast etc..) of the original image are preserved. Much work has been done on this topic during the past decade [Ashikhmin 2002; Durand 2002; Fattal 2002; Pattanaik 1998; Pattanaik 2000, Reinhard 2002; Rahman 1996; Schlick 1994; Tumblin 1999b; Ward 1997]. A detailed review on this topic is available from [DiCarlo 2001; Reinhard 2002]. We list some of the important algorithms in this section.

TUMBLIN'S LCIS [1999b] is most relevant to this research. Based on the observation that artists are able to create very realistic painting starting with the profile of the background and then adding details to it, Tumblin et al [1999b] introduced the novel idea of separating the detail of an image from its profile, range-compressing the profile to fit into the limited display range of LDR displays and finally adding the detail to generate realistic appearance. Their method makes use of anisotropic diffusion to extract details from the image at various spatial scales and makes use of a global non-linear compression operator for range compression of the profile. The only drawbacks of this method are that it generates excessive detail at times; uses at least 8 parameters in the detail generation process which makes the method a little formidable for user control. This may be the main reason why the method has not received wide popularity.

DURAND'S BILATERAL FILTERING [2002] **AND PATTANAIK'S ADAPTIVE FILTERING** [2002] techniques are also based on separating a HDR image into a base layer and a detail layer by bilateral filtering, and compressing the base layer. In both these techniques bilateral filtering is used to smooth the image while preserving the edges in the image. The resulting smooth image is used as the base layer.

The tone-mapping algorithm proposed in this thesis applies level set methods to separate the image into profile and detail. Compared to bilateral filtering method, a level set method based algorithm provides a function that allows the user to control the amount of detail. Compared to Tumblin's method, the parameters used in detail generation are fewer and are intuitive.

FATTAL'S GRADIENT DOMAIN TONE MAPPING ALGORITHM [2002] has some similarity to the above methods in that it computes the gradient field of the HDR luminance image and attenuates the gradient magnitudes. The LDR image is recovered by solving a Poisson equation on the modified gradient field.

WARD'S VISIBILITY MATCHING TONE REPRODUCTION OPERATOR [1997] converts the HDR image pixel intensities to JND (Just Noticeable Difference) units and applies the classical histogram equalization techniques. The operator simulates important perceived effects such as glare, color sensitivity and visual acuity.

REINHARD'S PHOTOGRAPHIC TONE MAPPING ALGORITHM [2002] leverages the practical techniques used in photographic practice, which maps the potential high dynamic range real world luminance to the low dynamic range of the photographic print.

RANGE COMPRESSION is one of the key steps in High Dynamic Range image display. Various functions, such as logarithm, square root and S-shaped (Sigmoid-function) functions have been proposed to carry out this compression. While all of these functions are approximations of the response of the visual system to the intensity of light, S-shaped function seems to be a preferable [Hunt 1995] representation. Various researchers [Schlick 1994; Tumblin 1999b; Pattanaik 2000, Pattnaik 2002; Reinhard 2002] used such a function for intensity range compression. The nice property of this function is that its range is bounded. This function plateaus at the higher and lower end of the intensity range, but gives linear response in about 2-3 log unit of intensity around the ambient intensity of the scene. Figure 2.1 shows a typical S-function. The major difficulty with an S-function is that it overly suppresses the detail in both high and low luminance areas of the image. As we see in Figure 2.1, the log luminance between $[2,+\infty)$ and $(-\infty,2]$ are compressed into a small response region $[0.88,1)$ and $(0,0.12]$, respectively. Thus the detail in regions of the image where log luminance is greater than 2 or less than –2, are compressed into a small region of 0.12. The larger is the luminance, the heavier is the penalty of the associated detail. Fortunately, the approaches based on separation of profile and detail are not affected because the details are separated before range compression and are added back detail after to the range compressed profile. Our method makes use of an S-function for profile compression.

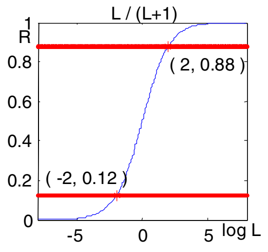


Figure 2.1: S-function

2.3.2 High Dynamic Range Image Data Compression

The raw size of an HDR image brings about problems in storage and transmission. Many bytes are wasted in data redundancy and perceptually unimportant information. This problem has been noticed by researchers and some preliminary algorithms have been proposed to compress them using encoding schemes such as RGBE, OpenEXR, LogLuv, etc.[Ward 2004a], followed by techniques such as RLE (Run Length Encoding) and LZW compression that exploit data redundancy. However, these approaches fall short of solving the problem, as compression using them provides us with only about a 50% reduction in size. Moreover, none of these provides a lossy compression alternative, as is popular in compressing standard images/videos. The work most related to this research addresses the HDR pixel encoding and proposes lossy HDR compression schemes as simple extensions of standard image/video codecs.

RGBE [Larson 1991] is one of the most popular HDR image formats so far. It converts a HDR color pixel data to a total of four bytes of data, three bytes for 3 channels of its color base, and one byte for the common exponent channel. In addition to its compact representation of 4 byte per pixel, it employs Run Length Encoding (RLE) to obtain on the average an additional 50% compression. Using the format high dynamic range data of 76 orders of magnitude can be represented at 1% accuracy. The range of this format is [1.5×10^{-39} 1.7×10^{-38}]. The accuracy of a pixel encoding method is computed as the percentage of unit incremental. The accuracy of RGBE is computed as $(1/2^8)/0.5=0.78\%$, which is approximated as 1%.

Ward proposed *LogLuv* format [Larson 1998] as a replacement of RGBE. It uses 16 bits for the logarithm of the luminance channel and 8 bits each for CIE (u', v') channels. This format can store 38 orders of magnitude at 0.3% accuracy. It also uses RLE to get an average 30% compression. Unfortunately, the change in color space to *Luv* from the commonly used color space of RGB has not made it popular among the HDR user community. The range of this format is [5.4×10^{-20} 1.8×10^{-19}], and the accuracy is computed as $2^{128/2^\wedge15}-1 = 0.27\%$, which is approximated as 0.3%. The 24 bits version of *LogLuv* format (*LogLuv24*) uses 10 bits for the logarithm of the luminance channel and 14 bits for the index of the chrominance channel in reference to a uniformly discretized visual gamut. *LogLuv24* can store 4.8 orders of magnitude at 1.2% accuracy.

ILM's *OpenEXR* [ILM 2004] is a recent HDR image format. It supports 16 bits float (*half*) for each channel. The *half* float uses 1 bit for sign, 5 bits for exponent, and 10 bits for mantissa. This format provides 10.7 orders of magnitude and 0.1% accuracy. It supports various compression schemes, like PIZ and RLE, and achieves a compression ratio of no more than 35%.

This format is adequate for storing HDR images of the real world because the range of illumination in the real world (from star light to sun light) spans approximately ten orders of magnitude [Reinhard 2002]. The range of this format is [6.0×10^{-8} 6.6×10^{4}] and the accuracy is computed as $1/2^{10}$=0.1%.

Image compression schemes such as JPEG and JPEG2000, and video codecs such as MJPEG and MPEG-4 are lossy compression algorithms that provide visually high quality images and videos [Gibson 1998; Dashti 2003; Poynton 2003; Richardson 2003; Sadka 2002]. Most image compression algorithms use transformation based image compression methods. JPEG compression uses a DCT based compression scheme. The original image is transformed into frequency domain using block wise (8x8 pixels per block) DCT transformation. The frequency components are quantized using a visual threshold based quantization table [Poynton 2003]. JPEG2000 [Rabbani 2002] differs from standard JPEG compression in that the raw image data is first transformed into the wavelet domain and the wavelet coefficients are then quantized. In both cases the quantized data are further encoded using adaptive arithmetic coding scheme. The data loss happens in the quantization step. In JPEG2000 the final compressed data are formed through a rate-distortion optimization operation to meet bit rate requirements. This step also introduces data loss.

Such compression schemes throw away visually unimportant information through a quantization step after a transformation that exposes the information redundancy in the images. The quantization step makes use of the deficiencies of the human visual system [Gibson 1998]. The video compression algorithms additionally use motion estimation/compensation to exploit temporal redundancy between video frames. Due to the robustness and proven success of these algorithms, it is desirable to build HDR image/video codecs around such algorithms.

Lossy compression schemes are necessary for aggressive compression. Methods based on lossy compression throw away information that is either not perceivable or is not significant to the HVS. The major issues in HDR lossy compression research are identification and separation of the visual unimportant information of HDR images. Very little work has been done on HDR image lossy compression and HDR video data compression. So far, we have come across only two lossy compression techniques that were developed parallel to our work. They are Ward's subband encoding [2004b] and Mantiuk's perception motivated HDR video encoding [2004].

WARD'S SUBBAND ENCODING METHOD [2004b] separates a HDR image into a tone-mapped LDR image and a ratio image, which is derived by dividing the original HDR image with the tone-mapped LDR image. The method makes assumption that tone-mapping compresses the intensity channel and hence the ratio image is a gray scale image. The tone-mapped LDR image is sent to a conventional JPEG encoder for lossy compression. The gray-scale ratio image is also compressed in JPEG encoder, and the compressed result is stored in a subband in the JPEG file of the compressed LDR image. The resulting compressed image allows direct display of the image in LDR display using conventional JPEG viewers. The HDR version of the image is reconstructed using the compressed image and the ratio image in the subband.

MANTIUK'S PERCEPTION MOTIVATED HDR VIDEO ENCODING [2004] is a hybrid encoding method, which applies compression to the HDR video in two stages, first to the luminance channel and then to resulting image in its frequency domain. The method first quantizes the luminance channel of the HDR video using a non-linear function, which distributes the quantization errors to match the perceptual threshold of the HVS in changing adaptation levels. It then sends the result from the first step to an extended DCT based MPEG-4 video encoder for further data compression in the frequency domain. In order to suppress the visual artifacts around the edges of sharp luminance changes, it supplements the MPEG-4 video encoder by extracting the edges around sharp luminance changes from the DCT blocks and compressing them separately using the Run Length Encoding (RLE) method.

We end this section by pointing out a couple of salient observations from HDR image compression research. First, human eyes have a log-linear response to intensities over a wide rage of intensities. Thus a constant or near constant quantization error in the logarithm domain is desirable. Second, visually unimportant information can be thrown away in HDR image compression. These observations have been utilized in designing most of the HDR pixel encoding such as the RGBE/XYZE and LogLuv [Larson 1998] formats, and in the two lossy image compression schemes discussed above. Our study makes use of this information to develop two HDR image/video compression methods [Xu 2005c, Xu 2005f]. Our method shares some structural similarity with Mantiuk's method, in the sense we do some quantization of the pixel data, and follow it by compression in a frequency space.

Having given the background material relevant to the research presented in this thesis we next proceed to describe our contribution in detail. We present each individual contribution in a separate chapter. Most of the presented work has been published in journals or conferences. Hence the content of most chapters is derived from those publications.

CHAPTER THREE: TONE MAPPING

This chapter presents a novel solution to the High Dynamic Range (HDR) image range compression problem using the level set framework [Xu 2003]. As discussed in the first two chapters, HDR images are the results of accurate rendering process. Range compression is important for visually accurate display of such images on ubiquitous low dynamic range display devices. It is also important that compression methods should be chosen such a way that many of the perceptual components are preserved. One such perceptual component is “detail” in the image. The method presented in this chapter attempts to preserve detail in a user controllable manner. Using level set framework, this method separates the HDR image into detail and profile. Then it uses a global tone mapping function to compress the profile. Finally it produces a low dynamic range version of the image for display by adding the details to the compressed profile. This method is capable of compressing the dynamic range while retaining the detail in the final image of various HDR images in a short time.

3.1 HDR Compression in Level Set Formulation

In the level set framework, an image can be considered as a surface composed of profile and details. The profile is the low frequency information and the detail is the high frequency information. The level set method is used to compute the profile of this surface. Detail in the image can be computed from the original image and the profile. We will first discuss the general compression process, then describe the level set method and finally give some experimental results.

3.1.1 General Compression Process

There are five steps used in the HDR image compression by our method. These are illustrated in Figure 3.1.

In our method, we compress only the luminance value, but keep the chrominance of each pixel. The original HDR image may take any format, like RGBE, TIFF, LogLuv, etc. We will first extract the luminance from each pixel, then compress them, and lastly recover pixel color from the compressed luminance. The method used to compute the luminance depends on the color space used to represent the original HDR image. For RGB color, we use the method by Greg Ward to compute the luminance, as shown in Equation (3.1).

$$I(R,G,B)=0.265R+0.670G+0.0648B \tag{3.1}$$

The log luminance values are processed by the level set method for profile computation. The computation technique will be discussed in detail on next subsection.

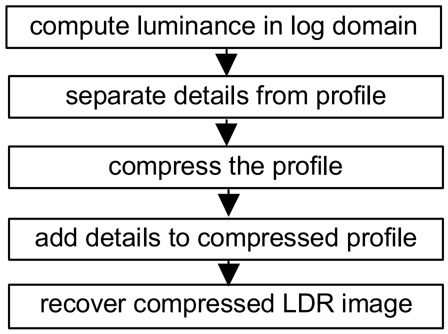

compute luminance in log domain
separate details from profile
compress the profile
add details to compressed profile
recover compressed LDR image

Figure 3.1: General HDR Compression Process

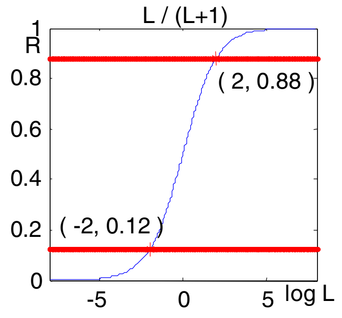


Figure 3.2: Sigmoid Function

After obtaining the profile from the luminance surface, we compute the details by subtracting profile from surface, as shown in Equation (3.2). Note that the subtraction in logarithm domain is equivalent to quotient before taking logarithm.

$$I_{\det ail} = I_{image} / I_{profile} \tag{3.2}$$

The profile is compressed using is the S-shaped function shown in Equation (3.3) [Pattanaik 2000].

$$R(I) = R_{\max} \frac{I^n}{I^n + \sigma^n} \tag{3.3}$$

where, I is pixel intensity or luminance, R is the compressed luminance ($0 < R < R_{\max}$) that is obtained by Equation (3.3), σ is the semi-saturation constant of I that causes the half-maximum response, and n is the sensitivity control parameter. Figure 3.2 is a plot of this equation with σ, $R_{\max}$ and n as 1. The value of $R_{\max} \leq 1$ maps the luminance values to a displayable range of $[0,1]$. The value of σ can be computed from ambient intensity values using equations proposed in [Pattanaik 2000]. For convenience, following [Reinhard 2002], we have taken the median of the luminance map divided by 0.18 as our σ value.

The compressed luminance map is finally recovered by adding the details to the compressed profile.

3.1.2 Computing Profile by Using Level Set Method

The level set method was first formulated by Osher et al. [Osher 1988] as a general method to manipulate the implicit surface. Ever since, it has been extensively used and studied as a powerful tool in various areas like physics, fluid mechanics, material sciences, computer vision, and computer graphics. We apply this method to extract the details from the luminance map.

The luminance map in the logarithm domain is denoted as a super-surface I over a 2-D domain. The motion of the surface is described by a partial differential equation (PDE) as shown in Equation (3.4).

$$I_t + F(\kappa)|\nabla I| = 0 \tag{3.4}$$

and the initial surface is determined by equation as shown in Equation (3.5).

$$I = I(x, y, t = 0) = 0 \tag{3.5}$$

where, I is the luminance value at pixel (x, y), t is the time, I_t is the derivative of I over time, $F(\kappa)$ is the changing rate of the luminance at pixel (x,y), κ is surface curvature at point (x, y), and ∇I is the first order partial derivation over x and y.

Since we want to compress the original logarithm luminance, we define our speed function as Equation (3.6).

$$\begin{aligned} F(\kappa) &= -e * \kappa \\ e &= \exp\left(-sen * \sqrt{\left|D_x^+ - D_x^-\right|^2 + \left|D_y^+ - D_y^-\right|^2} \right) \end{aligned} \tag{3.6}$$

where *sen* is the sensitivity to the edge, $D^{-x}, D^{+x}, D^{-y}, D^{+y}$ are first order forward and backward finite differences along the x and y axes respectively. Figure 3.3 shows the plot of the measure of "*edgeness*" as a function of the gradient change around pixels. "*edgeness*" here is used to describe the probability of finding an edge around pixels.

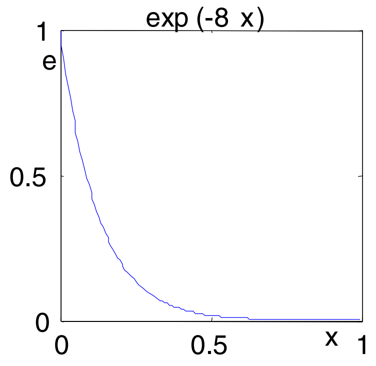


Figure 3.3: A Function Used for "edgeness"

Curvature faces towards the "protrusion", so the minus speed moves the surface inside, so as to suppress the "protrusion". e is used here as a measure of probability of an edge. The bigger of sen, the more sensitive the speed is to the edge. In our experiment, we use 8.0 for sen, and it works fine in all the test cases described later in this chapter.

The numerical solution to Equation 3.4 is shown in Equation (3.7) [Osher 1988].

$$I_{xy}^{n+1} = I_{xy}^{n} - \Delta t[\max(F_{xy},0)\nabla^{+} + \min(F_{xy},0)\nabla^{-}] \tag{3.7}$$

where, I_{xy}^{n}, I_{xy}^{n+1} are the luminance at point (x, y) at step n, $n+1$ respectively, Δt is the time step.

$$\begin{aligned}\nabla^{+} &= [\max(A,0)^2 + \min(B,0)^2 + \max(C,0)^2 + \min(D,0)^2]^{1/2} \\ \nabla^{-} &= [\max(B,0)^2 + \min(A,0)^2 + \max(D,0)^2 + \min(C,0)^2]^{1/2}\end{aligned} \tag{3.8}$$

$$\begin{aligned}
A &= D_{xy}^{-x} + \frac{\Delta x}{2} m(D_{xy}^{-x-x}, D_{xy}^{+x-x}) \\
B &= D_{xy}^{+x} - \frac{\Delta x}{2} m(D_{xy}^{+x+x}, D_{xy}^{+x-x}) \\
C &= D_{xy}^{-y} + \frac{\Delta y}{2} m(D_{xy}^{-y-y}, D_{xy}^{+y-y}) \\
D &= D_{xy}^{+y} - \frac{\Delta y}{2} m(D_{xy}^{+y+y}, D_{xy}^{+y-y})
\end{aligned} \tag{3.9}$$

and,

$$m(x,y) = \begin{cases} \begin{Bmatrix} x, |x| < |y| \\ y, |x| > |y| \end{Bmatrix}, xy \geq 0 \\ 0, otherwise \end{cases} \tag{3.10}$$

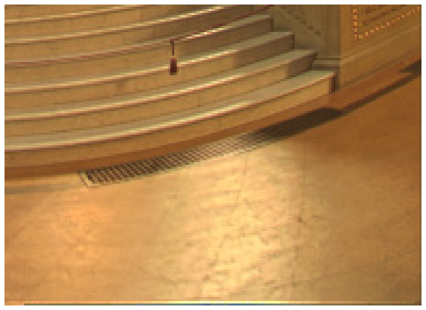
In Equation (3.9), $D_{xy}^{-x}, D_{xy}^{+x}, D_{xy}^{-y}, D_{xy}^{+y}$ are first order backward and forward finite difference along the x and y axes respectively. $D_{xy}^{-x-x}, D_{xy}^{+x-x}, D_{xy}^{+x+x}$, $D_{xy}^{-y-y}, D_{xy}^{+y-y}, D_{xy}^{+y+y}$ are second order backward, central and forward finite difference along x and y axis respectively.

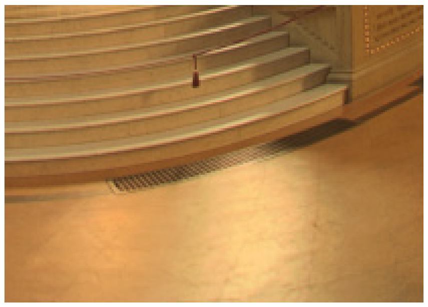
There are two problems in this numerical approach. First, Equation (3.7) can be solved iteratively, but how to determine when the iteration should be stopped? Second, how big is the Δt appropriate? If Δt is too small, more iterations are needed to get the same profile; vice versa, if too big, the computation will generate vibration and introduces noise. According to our experiments, it is found when the original luminance in logarithm domain ranges about 10, it is appropriate to use a value for Δt takes a value between 0.1 and 0.2. We map the original logarithm luminance into range [0, 10] before level set computation, and recover its original range after the computation. So, we can always use a value between 0.1 and 0.2 for Δt. The results shown in this chapter has been generated taking a value of 0.12 for Δt. As to the first problem, the more iteration, the more detail will be generated. We provide a parameter to the user to control the amount of detail. This is the only parameter of our approach, and images are shown to illustrate effects of this parameter in the next section. *Imgmanf* in our implementation is used to compute the profile using level set method. The Matlab 6.0 implementation of *imgmanf* is available in Table (3.1), and the whole algorithm for high dynamic range display is available in Table (3.2).

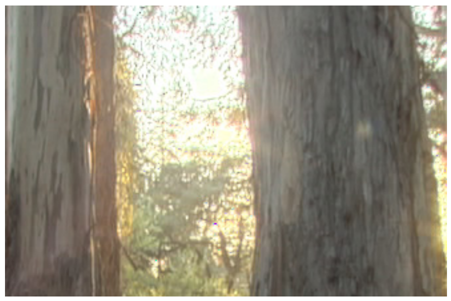
3.2 Experimental Results

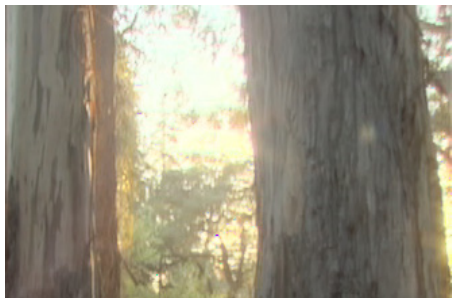
Some experimental results are shown in this section to verify our approach. Our method takes less than one minute for most of our test images on a 1.7G Pentium processor with 256M memory. The "memorial" image shown in Figure 3.4(a), which contains 768×512 pixels, took only 23 seconds on our platform. The results show more desirable detail in the final images, like highlights and texture. The images (a) and (c) of Figure 3.4 are the result from our algorithm, and the images (b) and (d) are results from applying the same S-shaped function to the whole image. The images (b) and (d) of Figure 3.4 are what one would get by using Reinhard's method [Reinhard 2002] without local adaptation. For Figure 3.4(a) and (b) (part of Memorial image), the floor texture is apparent in the image on Figure 3.4(a), but not apparent in the image on Figure 3.4(b). For Figure 3.4(c) and (d), you can also find many leaves between trunks in the image on Figure 3.4(c), which is lost in the image on Figure 3.4(d).

Figure 3.5 shows the tone mapping comparison between our level set method and Tumblin's LCIS method [1999b], Ward's visibility matching [1997], Reinhard's photographic method [2002], and Durand's bilateral filtering method [2002]. A visual comparison shows that our result looks better than the results from other methods. This may be because our method allows the user to tune to the most appropriate amount of details by an appropriate choice of parameter, which is missing in most other methods. The floor in Figures 3.5(c), (d) lacks details, while the floor in Figure 3.5(b) has excessive details. Of all the comparisons the best match in details is with those if Figure 3.5(e). More results using our method are available in Figure 3.8.


(a)


(b)


(c)


(d)

(a) and (c) are from Level Set based method. (b) and (d) are from S-compression alone.

Figure 3.4: Comparison of Tone Mapping Images

Our method has only one parameter, *nstep*, which controls how much detail in the final image. This parameter generally takes value between 5 and 50. The larger the parameter value, more the detail added to the final image. In our experiments, 15 was sufficient for most of the test images. Figure 3.6 shows the result as a function of the value of this parameter.

The image in Figure 3.6(a) contains the least detail, while image Figure 3.6(d) contains the most detail among the images in the figures. The four detail maps in Figure 3.7 correspond to the compression results in Figure 3.6 respectively. The size of detail map in Figure 3.7 is

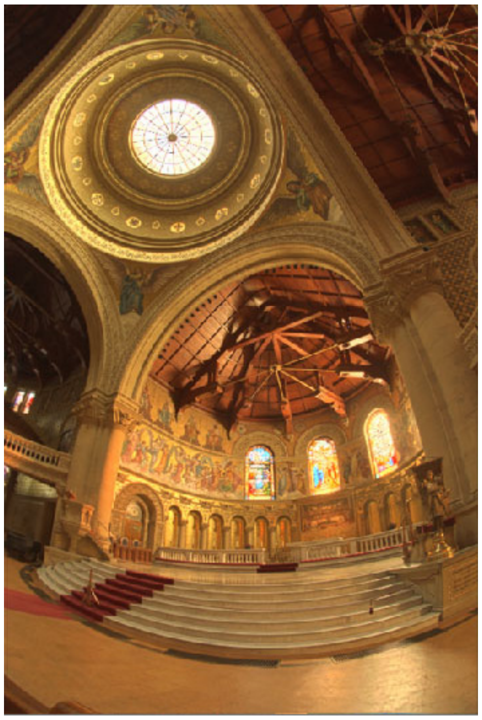
480×720, and the time to compute them on our platform (1.7G Pentium processor) are 0 secs, 4 secs, 7secs, and 13 secs, respectively.

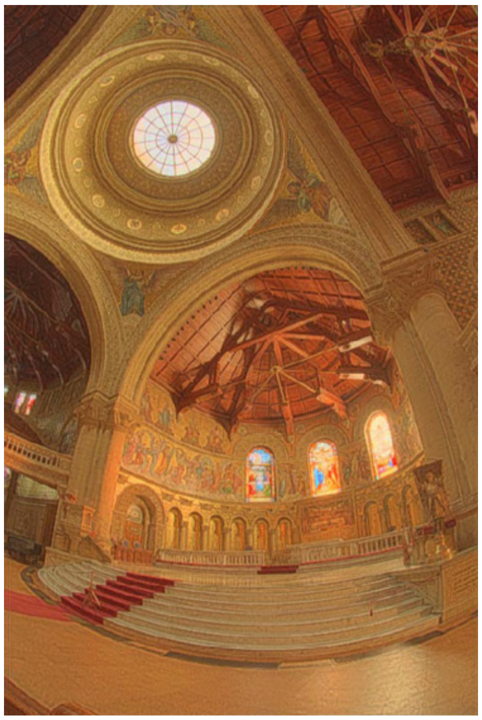
3.3 Analysis

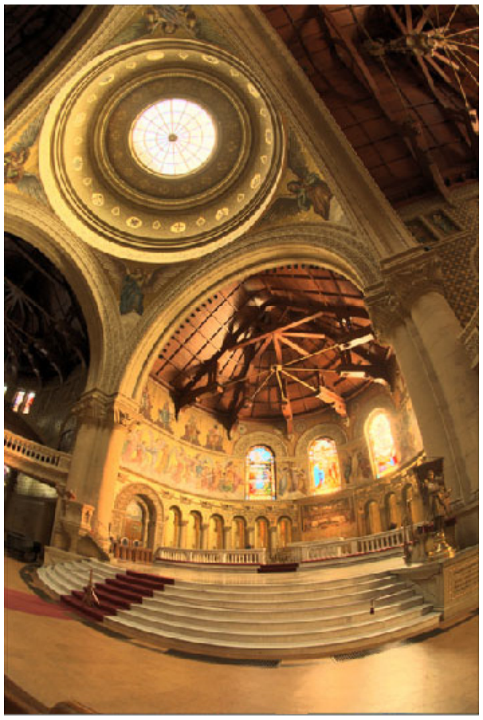
This chapter presents a High Dynamic Range image display method based on extraction of detail using level set method and the range compression function used in [Pattanaik 2000; Reinhard 2002]. The results are better than the application of a single range compression function on the whole image. The range compression function compresses the detail in the plateau area. However, this method avoids this problem by compressing the range in the profile image and restoring the previously extracted detail. This method is faster than the local adaptation based approach proposed [for example: Reinhard 2002]. The approach can be applied to most HDR images and obtain desirable results in one or two minutes.

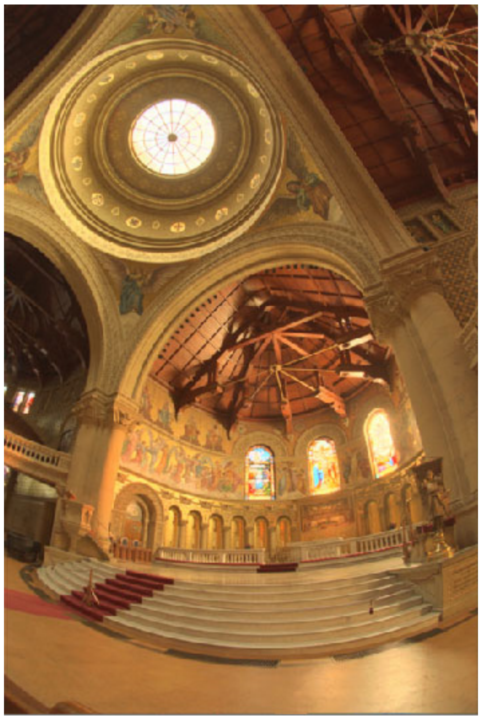
The fundamental approach proposed in our work, of separating profile and details, is also used in Tumblin's anisotropic diffusion based method [Tumblin 1999b] and in Durand's and Pattanaik's bilateral filtering based method [Durand 2002, Pattanaik 2002]. Our method and Tumblin's method are both iterative methods. However, profile extraction using Level Set method is simple, easier to control, and computationally more efficient than anisotropic diffusion used in [Tumblin 1999b]. Being a non-iterative method the bilateral filtering process is faster. Though it may not be a significant advantage, we would like to point out that the as compared to our method it lacks the versatility to control the amount of details. It must be noted that our work was developed in parallel to the development of bilateral filtering based method.

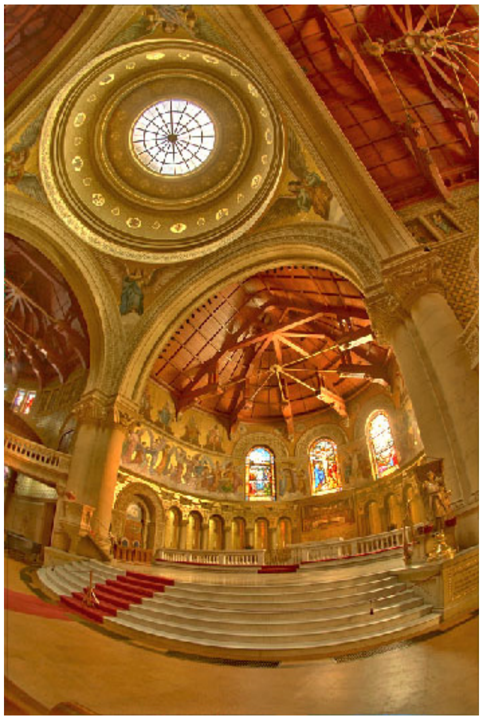
During our work, we also find some further questions. We use one parameter to control the amount of details. But, how much details will make the results look best? Too much detail or too little detail both spoils the final results. We think the solution is closely related to the perception of human vision. We also find noise sometimes exists with the details, and so one has to be careful not to accentuate the influence of the noise.


(a)


(b)


(c)


(d)

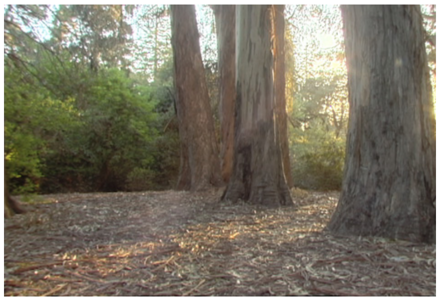

(e)

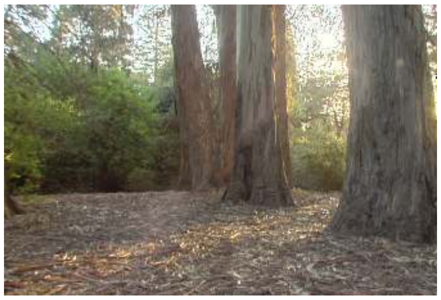
(a) our Level Set based method and; (b) Tumblin's LCIS method; (c) Ward's visibility matching method; (d) Reinhard's photographic method; (e) Durand's bilateral filtering method.

Figure 3.5: Comparison of Results of level Set Method and Other Methods

(a) nstep = 0

(b) nstep=3

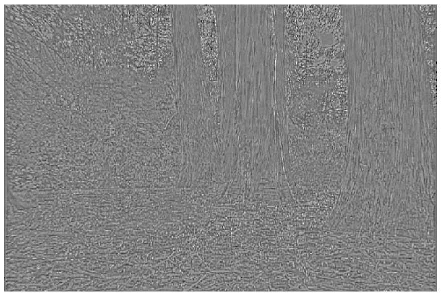

(c) nstep = 7

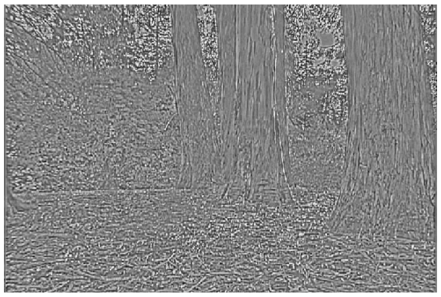

(d) nstep = 14

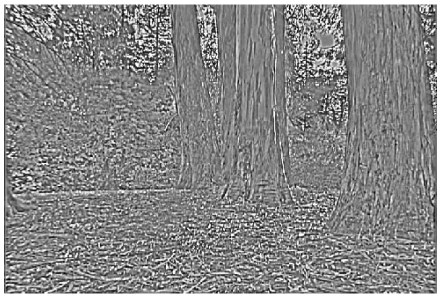
Images (a) to (d) have *nstep* as 0, 3, 7, 14, and Image (a) contains the least detail, and image (d) contains the most detail.

Figure 3.6: Images Taken With Different Parameter Value *nstep*

(a)

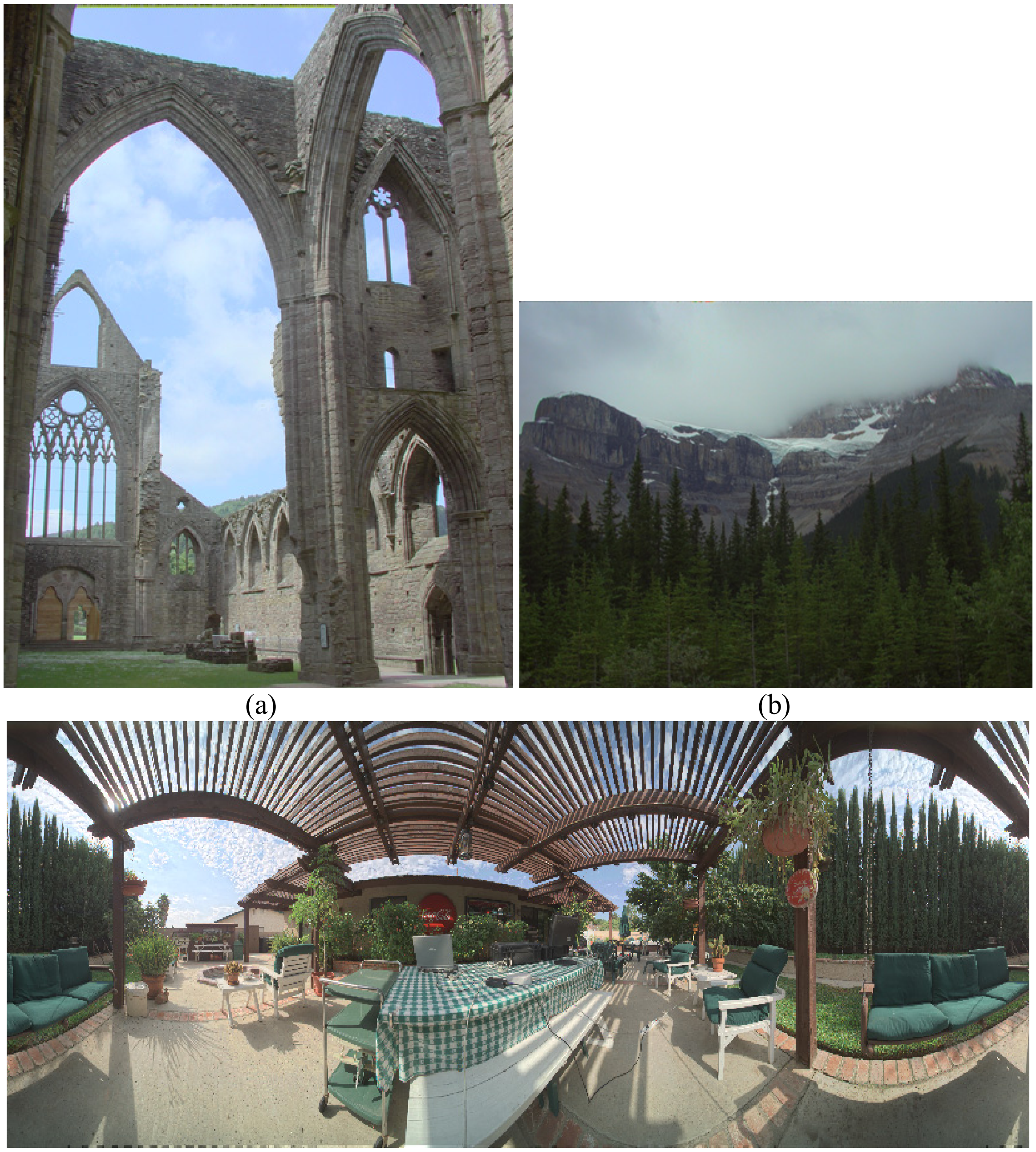

(b)


(c)


(d)

(a), (b),(c), and (d) show the details for *nstep*=0, 3, 7 and 14, respectively.

Figure 3.7: Details Extracted from "grove" with Level Set Method



The above images are compressed using our method. And the high dynamic range images (a),(b), can be found in http://www.cs.ucf.edu/~reinhard/cdrom/hdr.hmtl And (c) was kindly supplied to us by SpheronVR.

Figure 3.8: More HDR Range Compression Examples Using Level Set Method

Table 3.1: Matlab Code of Level Set Method for Computing the Profile

```
%%%this function implements Eqn 7.
function [rev]=imgmanf(im,delt,nstep)
%%%intialize vars
xdim = size(im,1); %size of image
ydim = size(im,2);
ll = 1;   rr = xdim; %boundaries of image
bb = 1; tt = ydim;
phi =  zeros(rr,tt);%local variables
nphi= zeros(rr,tt);
Npoint = (rr+tt)/2;
delx = 1/Npoint; dely = 1/Npoint;
eps = 1.0;
%%%Initializing original front
phi = init(im,ll,rr,tt,bb);
while nstep>0
   %%%compute new front
   nphi= phi_1(phi,ll,rr,tt,bb,delt,
      delx,dely,eps);
   phi = nphi;
   nstep = nstep - 1;
end
%%%return profile
rev = phi(ll:rr,bb:tt);
%%% initialize original level curve
function [phi]=init(im,ll,rr,tt,bb)
   for xx = ll:rr
      for yy =tt:bb
         phi(xx,yy) = im(xx,yy);
      end
   end
%compute curvature using central difference
function [rev]=curv(phi,j,k,ll,rr,tt,bb)
   phixx = (phi(j+1,k)- 2.0*phi(j,k) + phi(j-1,k  ));
   phiyy = (phi(j  ,k+1) - 2.0*phi(j,k) + phi(j  ,k-1));
   phixy = (phi(j+1,k+1) + phi(j-1,k-1) - phi(j-1,k+1)
           - phi(j+1,k-1))/4.0;
   phix = (phi(j+1,k  ) - phi(j-1,k  ))/2.0;
   phiy = (phi(j  ,k+1) - phi(j  ,k-1))/2.0;
%%%mean curvature
rev = (-phixx*(1.0+phiy*phiy)+2.0*phiy*phix*phixy-
phiyy*(1.0+phix*phix))/((1.0+phix^2+phiy^2)^1.5);
%%% solve PDE for one time step
function [rev]=phi_1(phi,ll,rr,tt,bb,delt,delx,dely,eps)
   newphi = zeros(rr, tt);
   for yy = bb:tt
      for xx = ll:rr
         cu = curv(phi,xx,yy,ll,rr,tt,bb);
         spd =  - eps*cu;
          if spd > 0.0 tg = g_HJ_plus(phi,xx,
                    yy,delx,dely,ll,rr,tt,bb);
         else  tg = g_HJ_minus(phi,xx,yy,
                        delx,dely,ll,rr,tt,bb);
          end
          newphi(xx,yy) = phi(xx,yy)  - spd*delt*tg;
       end
   end
   rev = newphi;
%%% compute numerical flux function
function [rev]=g_HJ_plus(phi,j,k,delx,dely,ll,rr,tt,bb)
   tA = A(phi,j,k,delx,dely); tB = B(phi,j,k,delx,dely);
   tC = C(phi,j,k,delx,dely); tD = D(phi,j,k,delx,dely);
   rev = -sqrt(1.0 + min(tA,0)^2+max(tB,0)^2
           + min(tC,0)^2+max(tD,0)^2);
%%% compute numerical flux function
function [rev]=g_HJ_minus(phi,j,k,delx,dely,ll,rr,tt,bb)
   tA = A(phi,j,k,delx,dely); tB = B(phi,j,k,delx,dely);
   tC = C(phi,j,k,delx,dely); tD = D(phi,j,k,delx,dely);
   rev = -sqrt(1.0 + min(tB,0)^2+max(tA,0)^2
           + min(tD,0)^2+max(tC,0)^2);
%%% compute derivative A of x at j,k
function [rev]=A(phi,j,k,delx,dely)
   DmDpPhi = phi(j+1,k) - 2.0*phi(j , k)  + phi(j-1,k);
   DmDmPhi = phi(j  ,k) - 2.0*phi(j-1,k)  + phi(j-2,k);
   rev = phi(j,k)-phi(j-1,k)
           + delx/2.0*m(DmDmPhi,DmDpPhi);
%%% compute derivative B of x at j,k
function [rev]=B(phi,j,k,delx,dely)
   DpDpPhi = phi(j+2,k) - 2.0*phi(j+1,k) + phi(j,  k);
   DpDmPhi = phi(j+1,k) - 2.0*phi(j,  k) + phi(j-1,k);
   rev = phi(j+1,k)-phi(j,k)
          - delx/2.0*m(DpDpPhi,DpDmPhi);
%%% compute derivative C of y at j,k
function [rev]=C(phi,j,k,delx,dely)
   DmDpPhi = phi(j,k+1) - 2.0*phi(j,k)  + phi(j,k-1);
   DmDmPhi = phi(j,k)   - 2.0*phi(j,k-1)+ phi(j,k-2);
   rev = phi(j,k)-phi(j,k-1)
          + dely/2.0*m(DmDmPhi,DmDpPhi);
%%% compute derivative D of y at j,k
function [rev]=D(phi,j,k,delx,dely)
   DpDpPhi = phi(j,k+2)-2.0*phi(j,k+1)+phi(j,k)  ;
   DpDmPhi = phi(j,k+1)-2.0*phi(j,k)  +phi(j,k-1);
   rev = phi(j,k+1)-phi(j,k)
         - dely/2.0*m(DpDpPhi,DpDmPhi);
%%% compute m for second order
function [rev]=m(x,y)
   if x*y>=0.0
       if abs(x)<=abs(y)  rev = x;  else   rev = y;   end;
   else   rev = 0.0;
end
%%%end
```

Table 3.2: Matlab Code of High Dynamic Range Image Display Using Level Set Method

```
function [ ]=lsHDRC(filename, nsteps)
%%%set up parameters
delt = 0.15; % Δt
%nsteps = 15; recommended
sensitivity = 8.0; %sen, for edge sensitivity
vision = 0.73;     % for vision, the smaller, the more
                   % accurate of the vision
CIE_rf=.265074126; CIE_gr=.670114631; CIE_br=.064811243; MIN_LUM = 1.0e-3;
%%%Read in the luminance image and store in r, g, b
[r g b]=rgbereadrgb(rgbefilename);
%%%Get luminance value
im = CIE_rf*r + CIE_gr*g + CIE_br*b;
im(find(im==0))=MIN_LUM;
%%% transform luminance into logarithm domain.
loglum = log(im);
%%% extract profile using LS    =>prologlum
prologlum = imgmanf(loglum,delt,nsteps,nnsteps,sensitivity);
meanlum = exp(median(loglum(:)));
n = vision;
factor = exp(prologlum).^(n-1) ./(exp(prologlum).^n
 + (meanlum/0.18)^n);
%%%recover low dynamic range image =>r,g,b
r = r .*factor; g = g .*factor; b = b .*factor;
%%%show compressed image
tiffwrite(r,g,b,lhrfilename);
%%%OK
```

CHAPTER FOUR: HIGH DYNAMIC RANGE IMAGE/VIDEO DATA COMPRESSION

Although high dynamic range (HDR) image and video are becoming common as rendering results from and as input to many computer graphics applications, their data compression has received little attention so far. In this chapter we propose two practical approaches to HDR Image/Video compression for efficient storage and fast transmission. Both of these approaches are extensions to existing image compression standards. The first one, a DCT based HDR image and video data compression method [Xu 2005f], is an extension to JPEG standard; and the second one: a wavelet based method [Xu 2005c], is implemented in JPEG2000 framework.

4.1 DCT-based HDR Image and Video Data Compression

Most current HDR images and videos are created from multiple exposures of the scene with varying exposures or varying f-stops. The most common HDR pixel representations such as RGBE and XYZE, use formats that are comprised of a base color and a common exponent. We compress the base color component by adaptive JPEG compression scheme. Since the common exponent takes a high weight in the pixel encoding, the quality of the HDR images is very sensitive to the noise introduced in the lossy compression of the common exponent. Hence we compress the common exponent using a lossless compression scheme. The strong coherence in the common exponent of most HDR image allows us a high compression rate even though it is compressed in lossless mode.

The primary contribution of our DCT-based HDR data compression method is the observation that high dynamic range image and video data are naturally separated into a base color component and common exponent component. The base color component may be thought of as a standard image/video, and hence existing image/video compression techniques can be used to compress this component. We suppress the introduction of artifacts by applying adaptive quantization coefficients. The common exponent component is compressed using a lossless mode. Because of the spatial coherence in the common exponent component, its compression ratio is very high and hence the sizes of the compressed HDR image/video are comparable to those of non HDR image/video. Thus we provide a simple, yet efficient approach to HDR image/video data compression.

This research define compression ratio as Equation (4.1).

$$\gamma = \left(1 - \frac{compressed\ size}{raw\ size}\right) \times 100\% \tag{4.1}$$

The rest of this section is organized as follows. The HDR image/video compression approach is first presented. Some experimental results, including the application of our HDR video compression technique, are then given in the next sub-section. Conclusions and future work are given in the final sub-section.

4.1.1 HDR Image and Video Compression

4.1.1.1 HDR Color Encoding

We have made use of RGBE color format for all our experiments. RGBE format is converted to/from raw HDR pixel format (3 floats for *r, g, b* channels) during coding/decoding. For the sake of completeness we give the conversions between floats and RGBE in Equations (4.2) and (4.3).

$$\begin{cases} v = \max(r, g, b) \\ m, e: \text{mantissa, exponent of } v \\ V = 256 \cdot m / v \\ R = r \cdot V \\ G = g \cdot V \\ B = b \cdot V \\ E = e + 128 \end{cases} \tag{4.2}$$

$$\begin{cases} m = 1/256 \\ e = E - 128 \\ v: \text{float with mantissa } m \text{ and exponent } e \\ r = (R + 0.5) \cdot v \\ g = (G + 0.5) \cdot v \\ b = (B + 0.5) \cdot v \end{cases} \tag{4.3}$$

where *m, e* are respectively the mantissa and exponent of a 32-bit floating point *v*, whose value is the maximum of *r, g* and *b* Throughout this chapter, we use R, G, B, E to denote the four 8-bit channels in RGBE format and *r, g* and *b*, to represent the three 32-bits floating point channels. This pixel format has successfully gone through the test of practical applications. It satisfies the visible color gamut requirement of most HDR images [Ward 2004a].

4.1.1.2 Compression Error Metrics

Several measures are introduced here to evaluate the efficiency and the quality of our HDR image/video compression methods. Compression time t_c and decompression time t_d are the (average) time to compress and decompress a frame, separately. Mean-squared-error (MSE) and peak-to-peak signal-to-noise ratio (PSNR) measure the objective compression quality, as shown in Equation (4.4).

$$MSE = \frac{1}{M \times N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left[x(i,j) - \hat{x}(i,j) \right]^2$$
$$PSNR = 10 \log_{10} \frac{R^2}{MSE} \tag{4.4}$$

where, *M, N* are the rows and columns of the image respectively, x and $\hat{x}$ are the original and reconstructed pixel luminance values and *R* is the image dynamic range (the difference between the maximum luminance and the minimum luminance). The value of MSE is inversely proportional to image compression quality; large values of MSE indicate poor quality. PSNR is a more natural error measurement, as its value is proportional to quality.

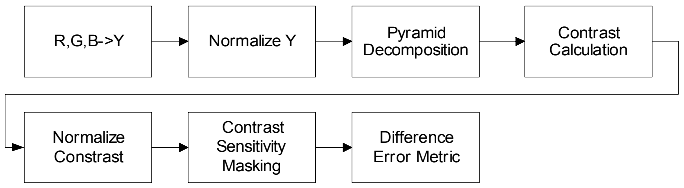
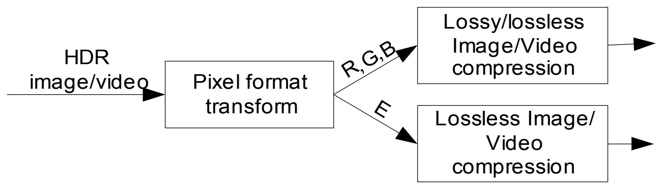


Figure 4.1: Simplified Sarnoff VDP

To measure the subjective compression quality, we make use of a simplified Sarnoff visual difference predictor (VDP) [Lubin 1995]. We first normalize the luminance channel to the range [0, 1] using a sigmoid function. Then, we compute the contrast images at different resolution levels and normalize these to the range [0, 1] using the same sigmoid function. Next, we apply to the outputs the contrast sensitivity function recommended in standard JPEG. Finally, we compute the squared difference between the test and reference images and sum them up across 3 levels. The square root of the summation is the perceptual error metric of each pixel. We use its mean value across the image as our VDP. The whole process is shown in Figure 4.1. Large value of VDP means low compression quality, and vice versa.

4.1.1.3 DCT-Based HDR Image Compression

The general scheme of HDR image/video compression is illustrated in Figure 4.2.


HDR image/video
Pixel format transform
R,G,B
E
Lossy/lossless Image/Video compression
Lossless Image/ Video compression

R, G, B, i.e., color base, and E, i.e., common exponent, are separated and sent to different compression schemes.

Figure 4.2: HDR Image and Video Compression Scheme

Compared to lossy compression, HDR lossless compression is a simpler process wherein the color base and common exponent are directly compressed using some entropy coding method, like CABAC from H.264/MPEG-4 (Part 10) standard [Marpe 2001], which is an arithmetic coding scheme with excellent compression performance.

Our lossy compression scheme is as follows. Inspired by the steps leading to JPEG compression, we introduce a color space transform and subsampling step, followed by a DCT transformation step and a quantization step before entropy coding. This allows us to expose and throw away visually unimportant information in the color base, achieving a higher compression ratio, as shown in Figure 4.3. The common exponent is directly sent to entropy coding.

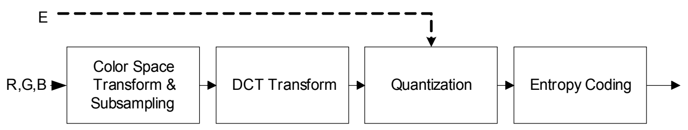


Figure 4.3: HDR Image Lossy Compression of the Color Base

The observations and algorithms of conventional 24-bit RGB images also apply to the color base of HDR images, except the quantization step. The R, G, B is first transformed to a luminance/chrominance space, YC_bC_r. Since human eyes are less sensitive to chrominance than luminance, the chrominance can lose more information and still be less noticeable. So, the chrominance can be sub-sampled. The DCT transform step converts the 8 by 8 blocks in luminance/chrominance channels into frequency space for information loss in later quantization step. The quantization step divides each frequency coefficient from the last step by its quantization coefficient (QC) and then rounds the outcome to an integer, which is finally sent to entropy coding. For the viewing condition similar to that of conventional 24-bit RGB image, we use the QC from JPEG Standard as our HDR image QC, as shown in Equation (4.5).

$$Q_Y = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix} \quad Q_{C_b,C_r} = \begin{bmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 47 & 66 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{bmatrix} \tag{4.5}$$

The pixels in one DCT block may have different exponents. The pixel with higher common exponent will suffer more information loss, which is proportional to $2^{\Delta E}$, where ΔE is the common exponent difference. To compensate the accuracy loss of pixels with higher common component in one 8 by 8 block, it is necessary to divide Q_Y and Q_{CbCr} with some compensation coefficient q_c, as shown in Equation (4.6).

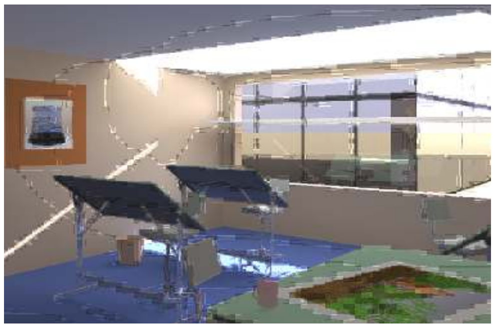
$$q_c = 2^{\max\{E_i\}-\min\{E_i\}}, i = 1,...,64 \tag{4.6}$$

In our experiments, we found the compensation coefficients in Equation (4.7) provide better performance. Equation (4.7) is a linearly increasing function, which has a less steep slope than Equation (4.6).

$$q_c = \left\lfloor \frac{\max\{E_i\} - \min\{E_i\} + 3}{2} \right\rfloor \tag{4.7}$$

The QC is selected adaptively based on the dynamic range one block covers, and this compression method is actually adaptive JPEG (AJPEG) [Rosenholtz 1996].

The E channel represents the common exponent of the pixel color. Note that a unit error in the E channel gives rise to a significant error in the decompressed HDR image. Figure 4.4 highlights this problem. To avoid this, we apply lossless compression to the E channel.



The error from E channel brings about obvious artifacts in the recovered HDR image. Most of the error is around the region boundaries of the E component.

Figure 4.4: Lossy Compression of E Channel

The R, G, B channels are compressed in either lossless or lossy mode. We experimented with two different modes of HDR image compression:

HDR image lossless compression:

- Color base: lossless (CABAC, LJPEG, etc.)
- Common exponent: lossless (CABAC)

HDR image lossy compression:

- Color base: lossy (AJPEG, etc.)

- Common exponent: lossless (CABAC)

In both the modes, the color base is compressed as a conventional color image, and E can also be compressed as a conventional grayscale image. Figure 4.5 shows an HDR image compressed in different modes.

4.1.1.4 DCT-Based HDR Video Compression

We extend the HDR image compression method to HDR video compression. We use only lossless video compression schemes for the common exponent channel. Thus, the two different modes for HDR video compression are:

HDR video lossless compression

- Color base: lossless (CABAC, etc.)
- Common exponent: lossless (CABAC)

HDR video lossy compression

- Color base: lossy (MJPEG, MPEG, etc.)
- Common exponent: lossless (CABAC)

For lossless compression of HDR video, arithmetic encoding can be used to compress the color base and common exponent separately. The lossy mode compresses the intra frame (I frame) using the HDR image compression approach described in the last subsection. The motion vector (MV) from motion estimation of the inter frame is sent directly to entropy coding [Richardson 2003]. A simplified scheme is depicted in Figure 4.5. The residue of the P, B frames is also coded using the HDR image compression approach.

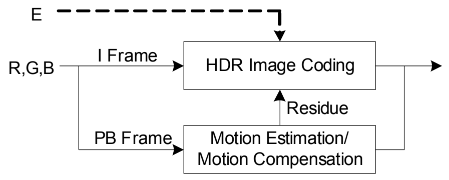


Figure 4.5: HDR Video Lossy Compression of Color Base

Based on the conventional video codec, the HDR video codec adjusts the QC of each block using q_c of Equation (4.7).

4.1.2 DCT-Based Experimental Results

Our HDR image/video compressor saves significant amounts of storage by ignoring perceptually unimportant information, a process that is illustrated by experiments.

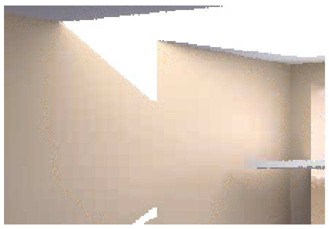
Figure 4.6 shows some results from our HDR image compression algorithm using various compression schemes. Figure 4.6(a) is the original image. Higher compression ratios are achieved at the expense of quality, which is depicted by larger MSE/VDP and smaller PSNR values. The captions in Figure 4.6(b) to (d) give the compression schemes followed by the quality level of lossy compression. The compressed image quality is visually acceptable up to a quality level of 16, and “block” artifact inherent to DCT appear at low quality levels. In Table 4.1 we tabulate statistics for these images. We see the expected increase in MSE/VDP and decrease in PSNR as the compression ratio increases. The t_c and t_d are almost independent of compression quality level. Compared to RGBE and OpenEXR, our HDR image lossy compression uses 1/10 the storage with little visual quality degradation. Our method is fast while achieving high compression ratios and quality levels for HDR images.

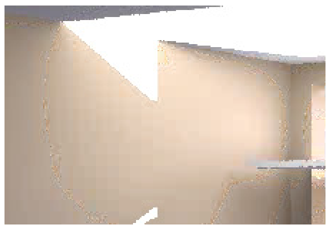
(a) Original

(b) AJPEG+CABAC

(c) AJPEG+CABAC, 16

(d) JPEG+CABAC, 16


(e) subimage of (c)


(f) subimage of (d)

The original image is shown in (a). (b)-(d) show compressed images using different compression quality, which has 32 levels from the highest 0 through the lowest 31. "AJPEG+CABAC, 0" represents compressing base color with adaptive JPEG compression (compression quality=0) and common exponent with CABAC compression. Subtitles (c) and (d) can be interpreted similarly. Noticeable jaggies appear in images compressed using "JPEG+CABAC, 16", compared with the artifacts removed by "AJPEG+CABAC, 16".

Figure 4.6: Lossy and Lossless Compression with Different Compression Qualities

Table 4.1: Statistics to Figure 4.6 (a)-(d)

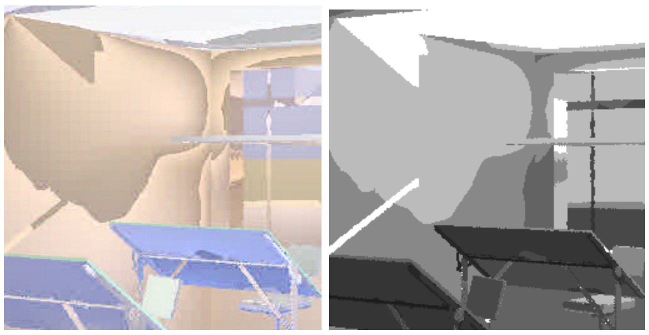
	Size	t_c	t_d	MSE	PSNR	VDP
	KB	sec.	sec.	10^{-3}	dB	10^{-3}
RGBE	786.0,69.0%	n/a	n/a	n/a	n/a	n/a
OpenEXR	701.0, 71.0%	n/a	n/a	n/a	n/a	n/a
CABAC+CABAC	181.0, 92.5%	0.36	0.130	n/a	n/a	n/a
LS-JPEG+CABAC	269.0, 89.0%	0.30	0.090	n/a	n/a	n/a
AJPEG+CABAC,0	123.0, 95.0%	0.38	0.047	0.13	65.2	.59
AJPEG+CABAC,16	78.5, 96.8%	0.41	0.046	0.62	58.5	1.1
JPEG+CABAC, 16	51.0, 97.9%	0.38	0.046	1.85	53.7	1.8

The HDR image is of size 800 by 754, and has a dynamic range of [0.001, 20.875]. These statistical data and all others are collected on a Dell computer with Intel Xeon 2.4G CPU and 1G memory running Windows XP.

The lossless “CABAC+CABAC” compression has obvious higher compression ratios than the simple RLE encoding proposed in the original RGBE format, but for more aggressive compression of HDR image, lossy compression is necessary.

Figure 4.6(f) shows the artifacts when compressing color base without our adaptive JPEG scheme. These artifacts mainly appear as jaggies around the region boundaries of the common exponent channel, as shown in Figure 4.7(b). The reason is that new edges are formed in the color base channels around the region boundaries of the common exponent channel, as shown in Figure 4.7(a). These new edges introduce the jaggies artifacts into the compressed image. It is important to suppress these artifacts by applying smaller quantization coefficients for less quality loss around these edges using our method.

We rendered a sequence of HDR images of the scene “room” [Ward 2005]. This sequence “room” consists of 45 frames, each of size 800 by 574 in RGBE format and occupies 32.8M. We compressed it in “CABAC+CABAC” mode and “AMJPEG+CABAC” mode with various compression qualities for comparison purposes.



(a) R,G,B channels (b) E channel

New edges formed on (a) around the edges on (b).

Figure 4.7: Separation of R,G,B Channels and E Channel

Table 4.2: Some Statistics of Image “room”

	Total size	$\bar{t}_c$	$\bar{t}_d$	$\overline{MSE}$	$\overline{PSNR}$	VDP
	MB	sec.	sec.	10-3	dB	10-3
CABAC+CABAC	7.26, 91.0%	0.52	0.12	n/a	n/a	n/a
AMJPEG+CABAC,0	4.89, 93.9%	0.41	0.041	0.11	60.3	.56
AMJPEG+CABAC,16	2.96, 96.3%	0.41	0.042	0.66	52.6	1.2
MJPEG+CABAC,16	2.01, 97.5%	.39	0.042	1.54	49.0	1.9

Table 4.2 shows the compression statistics. The decompression speed is about 25 fps. This table clearly shows the effectiveness of our HDR video compression approach. The compression quality conflicts with compression ratio, but the compression and decompression time is independent of compression ratio.

In all the video compression examples, the common exponent channel of each frame is compressed independently using a lossless CABAC algorithm. More compression can be achieved if inter-frame temporal coherence of the E channel is exploited.

Among all existing common image/video codecs, it is hard to say which one is the best. We provide a general framework and guideline for HDR image/video compression based on normal image/video compression. Our method is practical and takes benefits of the established and well tested normal image/video compression schemes.

The choice of HDR image/video compression quality levels depend on its possible down-stream applications. Lossy compression of HDR image/video of medium quality level seems the best compromise between compression ratio and compression quality.

The HDR image and video examples are available in http://graphics.cs.ucf.edu/hdri/index.php#video. More examples are shown in Figure 4.9.

4.1.2.1 Application to Dynamic Object Real-Time Rendering

We applied HDR video to rendering of dynamic objects in real time [Xu 2004]. For each frame of the animation, we calculate its pre-computed radiance transfer (PRT) coefficients [Sloan 2002] for a number of spherical harmonics basis environment lights. We unfold the dynamic object surface to a 2D map. Each pixel belonging to the unfolded map stores PRT data corresponding to its surface point. The resulting map is called a spherical harmonics light map (SHLM). A sequence of SHLM corresponds to an animation sequence. Each pixel on the SHLM uses floating point numbers to keep both its dynamic range and accuracy. The size of the SHLM S depends on the sampling rate γ over the object surface and its area Φ, as shown in Equation (4.8)

$$S = \gamma \cdot \Phi \tag{4.8}$$

In our application we use an animated human model with a surface area of 5 square meters. At 10 samples per square centimeter, and 25 spherical harmonics coefficients, each SHLM map occupies 50MB. Thus a sequence of 100 frames occupies 5GB. This high volume of this data is difficult to store and use in real-time.

Using the "AMJPEG+CABAC, 16" scheme we compress each SHLM individually. At the rendering time we decompress SHLM in CPU. The overhead of decompression time is trivial compared to the time required to load data from CPU to GPU memory. It is worth noting that video codec hardware is also available for faster HDR video codec.

We have computed a warrior walk animation of 100 frames. With a single SHLM size of 128 by 128, the SHLM raw data is over 100M. We are able to reduce its size to 4.2M with good compression quality, and 3.5M with reduced but acceptable compression quality.

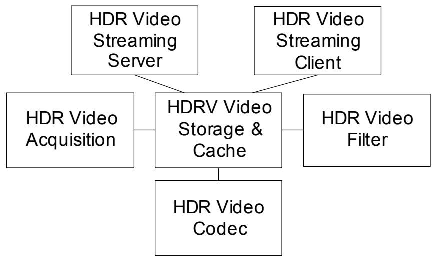
4.1.3 Analysis of DCT-Based HDR Data Compression

It is shown that the HDR image and video can be efficiently compressed by applying standard image/video compression algorithms separately to the base color and common exponent components. We restrict the common exponent channel compression to be lossless.

Our approach is simple and efficient, and it is still possible to improve it in various ways. The temporal coherence of common exponent can be further exploited for higher compression ratios by applying predictive encoding. It is also possible to use wavelets instead of DCT in the transformation step, since wavelets are reported to have a better compression performance [Salomen 2000].

However, the non-linear treatment of HDR color may bring about several problems. Some close values (e.g., $128x2^8$ and $255x2^7$) become far from each other in the RGBE format (128 and 255). Additionally, our approach is limited to compressing positive color values. We currently overcome the first drawback by applying higher quality compression. Moreover, it is possible to compress the common exponent part in lossy mode for additional compression. Finally, the correlation between the common exponent and base color channels can be exploited for further compression.

The basic approach in this chapter can also be adapted to compress non-image data. Terrain data is often represented as an elevation function on a 2D mesh. The terrain data used in real applications are often high volume. This kind of terrain in digital elevation mesh (DEM) format can be seen as a high dynamic range image and compressed. Human vision based compression components of JPEG must be re-evaluated here, though.


HDR Video Streaming Server
HDR Video Streaming Client
HDR Video Acquisition
HDRV Video Storage & Cache
HDR Video Filter
HDR Video Codec

HDR video codec is the basis of other HDR video processing components.

Figure 4.8: A General HDR Video Manipulation Framework

In remote walkthrough of virtual environments, HDR image/video compression plays a key role. The HDR textures and HDR environmental lighting should be compressed before transmission. By making use of the underlying normal image/video compression/streaming techniques, progressive HDR image/video streaming is simple and efficient.

Light field data are often of high volume. Our work provides a new possible way for its compression. The light field data can be unfolded to 2D space, and then compressed by exploiting the information redundancy between neighboring vertices. We leave this as part of our future research.

HDR video codec underlies other applications of HDR video. These relations are apparent in Figure 4.8.

Although some work has been done about HDR video acquisition, much of this area remains unexploited. We hope our work will inspire more research in this area.

(a) “BigI_big.hdr” (3720x1396) 10,567K in RGBE format

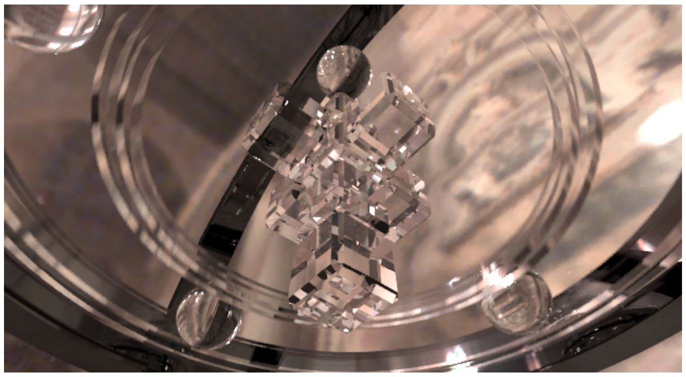
(b) “BigI_big.hdr” 1,234K in “AJPEG+CABAC, 16” format

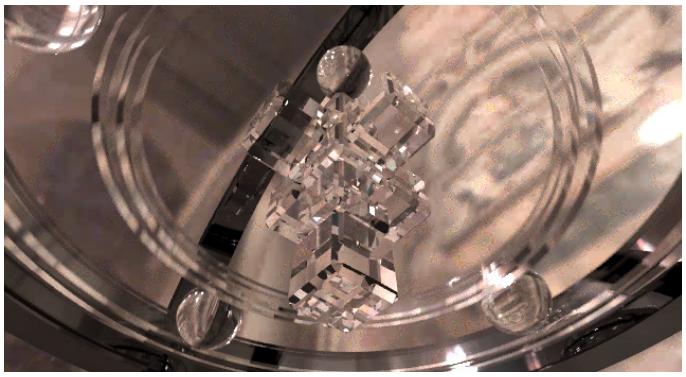
(c) “park.hdr” 1,116K in RGBE format

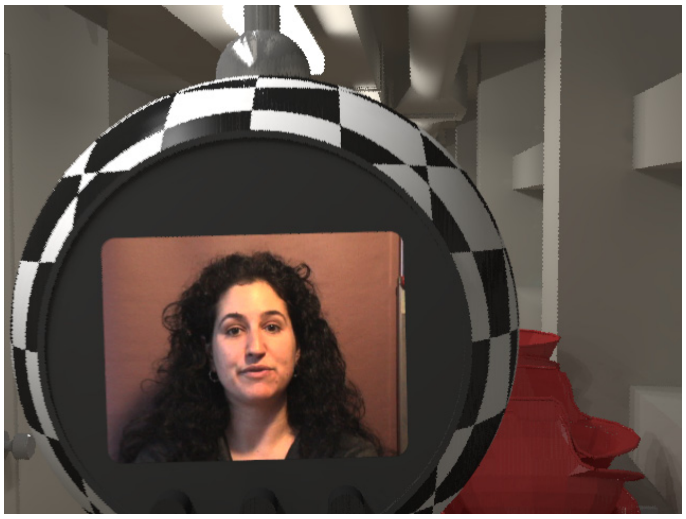
(d) “park.hdr” 91K in “AJPEG+CABAC, 16” format

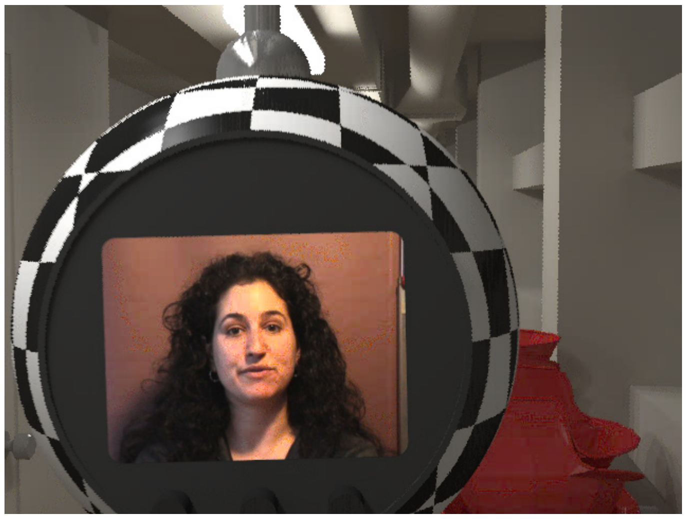
(e) “tunnelnosunB” 443Min RGBE format, 900 frames

(f) “tunnelnosunB” 39.3M in “AMJEPG+CABAC, 16” format


(g) “cross” 270M in RGBE format, 390 frames


(h) “cross” 39.1M in “AMJPEG+CABAC, 31” format


(i) “DetailAttentionSeg0” 652M in RGBE format, 869 frames


(j) “DetailAttentionSeg0” 56.4M in “AMJPEG+CABAC, 16” format

(a)-(d) show compression of two HDR images, and (e)-(j) show compression of three HDR videos. (c)-(f) are courtesy of Pattanaik, (g)-(h) are courtesy of Debevec, and (i)-(j) are courtesy of Ward.

Figure 4.9: More DCT-Based HDR Data Compression Examples

4.2 Compression Scheme in JPEG2000

This section examines the compression scheme used in JPEG2000 [Rabbani 2002]. The raw image data are first transformed into the wavelet domain and the wavelet coefficients are then quantized. The quantized coefficients are encoded using adaptive arithmetic coding. The final compressed data stream is formed through a rate-distortion optimization operation to meet bit rate requirements. The whole process is briefly shown in Figure 4.10.

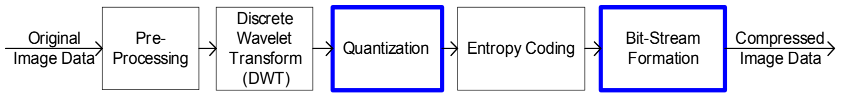


Figure 4.10: JPEG2000 (Part 1) Fundamental Building Blocks

The information loss happens in the two stages bordered in blue in Figure 4.10. The pixel bit depth is first reduced in the quantization step [ISO/IEC 2000], where knowledge of the human visual system may be applied. The compressed data are then truncated in the bit-stream formation stage, where a minimum distortion is enforced within the bit rate budget.

4.2.1 Approach to HDR Still Image Compression

The overall HDR compression/decompression scheme is shown in Figure 4.10. There are two basic components in this scheme: pixel encoding/decoding and image encoding/decoding. The blocks bordered in green are the steps we introduce for pixel encoding. The blocks bordered in black, "JPEG2000 Encoder/Decoder," are the standard JPEG encoding/decoding schemes shown in Figure 4.10, and use our new quantization steps for HDR image lossy compression. The following paragraphs briefly describe each of the steps of our scheme.

The raw HDR image data are transformed into the logarithm domain, and then uniformly quantized into n bits using the following equation.

$$[\bar{r},\bar{g},\bar{b}] = f([r',g',b']:n) \tag{4.9}$$

where

$$\begin{aligned}[r',g',b'] &= \log([r,g,b]) \\ f(x:n) &= \left[(x - x_{\min})/(x_{\max} - x_{\min})\cdot(2^n - 1)\right]\end{aligned}$$

r, g, b are the raw colors represented using three 32-bit floats in RGB color space, *r', g', b'* are logarithms of *r, g, b* respectively, and $\bar{r}, \bar{g}, \bar{b}$ are the colors represented in unsigned integers of n bits. x_{min} and x_{max} are the minimum and maximum value of each channel in the logarithm domain. We use floating point numbers in logarithmic transformation, thus we have only trivial, if any, data loss in this transformation. The time consumed is acceptable for single HDR image encoding/decoding and can be improved using a GPU implementation.

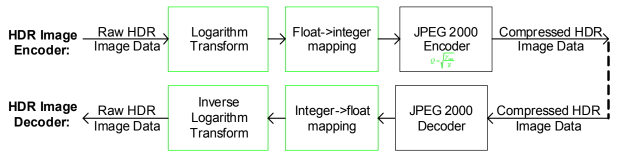


Figure 4.11: HDR Image Compression and Decompression Diagram

The pixel encoding scheme plays an important role in preserving the color gamut and dynamic range of original raw HDR images. Our simple encoding scheme of mapping raw pixel values in three 32-bit floats into those in three n-bit integers, as shown in Equation (4.9), keeps the original color gamut and dynamic range, with the expense of introducing a coding error in the logarithm domain. The value of this error is shown in Equation (4.12). Our method takes a non-negative RGB color space, whose color gamut covers the most commonly used colors. This constraint is common with most HDR images available to the computer graphics community.

We send the image in unsigned integers resulting from pixel encoding to the JPEG2000 encoder for image compression. We enable the standard color transformation option available in the JPEG2000 encoder to take advantage of color decorrelation (see Annex G of [ISO/IEC 2000]). This transforms the color linearly from logarithmic RGB space to YC_bC_r space. The color transform over logarithmic RGB operates in a non-linear domain, which possibly leads to luminance/chrominance mixing to some degree. We thus disable the chrominance subsampling, which depends on luminance/chrominance separation and is used in LDR image encoding.

The image data in YC_bC_r space are then transformed to wavelet space. We quantize each subband b of the wavelet transformation using a quantization step Δb computed using Equation (4.10).

$$\Delta b = \sqrt{\gamma_{\max} / \gamma_b} \tag{4.10}$$

where γ_b is the energy weight for subband b, defined as the square of the amount of error introduced by a unit error in the transformed coefficient (see Annex E.2 of [ISO/IEC 2000]). $\gamma_{\max}$ is the maximum energy weight of all subbands.

This quantization scheme is different from the JPEG2000 standard recommendation (see Annex J.8 of [ISO/IEC 2000] and is chosen to maintain displaying and viewing conditions independence by removing the perception related factor. HDR image formats are scene referred as opposed to LDR formats, which are image referred.

The quantized result is then transformed into bit streams through entropy encoding. And the bit stream is finally truncated to the desired bit rate through the rate control mechanism in JPEG2000. The rate control is implemented by rate-distortion optimization, which must satisfy the bit rate constraint while minimizing the distortion, i.e., the MSE of the reconstructed image in the logarithm domain.

For decompression, the compressed HDR image data are first decoded using a JPEG2000 decoder, and the results are then converted to raw HDR image data via the inverse operation of Equation (4.9). The equations are as follows.

$$\left|\overline{\overline{r}},\overline{\overline{g}},\overline{\overline{b}}\right|=f'([r'',g'',b'']:n) \qquad (4.11)$$

where

$$[r'',g'',b'']=\exp([\overline{r},\overline{g},\overline{b}])$$
$$f'(x:y)=x/(2^{y}-1)\cdot(x_{\max}-x_{\min})+x_{\min}$$

And the parameters xmin and xmax in Equation (4.13) are the same as those in Equation (4.9).

The error is analyzed in detail to show its sources during the encoding process.

4.2.2 Pixel Encoding Precision

The parameter n is left to the user to manually control the coding error ε_c in the logarithm domain, arising from the quantization step. ε_c can be expressed using Equation (4.12).

$$\varepsilon_c=(x_{\max}-x_{\min})/(2^{n+1}-2) \qquad (4.12)$$

To restrict the maximum coding error in the logarithm domain to ε, we must use n, determined using the following equation.

$$n=\lceil \log_2((x_{\max}-x_{\min})/\varepsilon+2)-1\rceil$$

Thus for a dynamic range of 12 orders of magnitude, and value of n equal to 16, the coding error in the logarithm domain is $12/(2^{16+1}-2)=0.01\%$.

When comparing to other coding methods it is convenient to convert the coding error in the logarithm domain to a relative error E using Equation (4.13). The relative error of some coding scheme is defined as the ratio of the difference to the smaller value of two consecutive codes in the coding scheme.

$$E=10^{2\varepsilon_c}-1 \qquad (4.13)$$

4.2.3 Error Sources

There are three operations that may introduce error in the encoding process: conversion of float to integer, coefficient quantization, and rate-distortion optimization.

The error in lossless mode is limited only to the float conversion error.

$$\varepsilon_{lossless} = \varepsilon_c \tag{4.14}$$

The error in lossy mode, ε_{lossy}, is the sum of the pixel encoding error ε_c, the coefficient quantization error, ε_q, and the bit stream truncation error, ε_{r-d}.

$$\varepsilon_{lossy} = \varepsilon_c + \varepsilon_q + \varepsilon_{r-d} \tag{4.15}$$

The coefficient quantization, ε_q, is defined as:

$$\varepsilon_q = \sum_b \Delta b \cdot \sqrt{\gamma_b} \tag{4.16}$$

where Δb is the quantization step for subband b, and ε_{r-d} is defined by Equation (4.19).

$$\varepsilon_{r-d} = \sqrt{\sum_i D_i^*} \tag{4.17}$$

where D_i^* is the distortion of codeblock *i* after rate-distortion optimization (see Annex J.10 of [ISO/IEC 2000]).

4.2.4 Compression Results

We implement our HDR compression scheme as an extension to the JasPer API (C implementation of JPEG2000 part 1) [Adams 2000]. Figures 4.12-4.14 show the compression results using our HDR image compression scheme. The maximum n supported by JasPer is 16. Hence, we always use n=16, and use the R-D optimization of JPEG2000 to automatically reduce the compressed data to a desired bit rate. It is possible that some raw pixel values are zero. This poses a problem for conversion of an image into the logarithm domain. We overcome this problem by replacing those pixels values with the minimum non-zero channel value.

4.2.4.1 Lossless Mode

In Table 4.3 we show the comparison statistics of our lossless compression scheme with other existing lossless schemes. For all four test HDR images, our compression scheme performed poorly compared to all others. We hypothesize that this is because the JPEG2000 compression scheme is not designed for lossless compression.

Table 4.3: Storage requirements of different HDR image formats

	Relative error.x10^{-3}	"BigFogmap"	"Memorial"	"park"	"tahoe"
Dynamic Range		4.2	5.9	3.6	4.8
Our Lossless	see notes	3.3M	1.6M	1.3M	11.6M
RGBE(RLE)	10	2.7M	1.3M	1.1M	10.9M
LogLuv(32 bits)	3	2.5M	1.2M	0.7M	7.1M
OpenEXR(PIZ)	1	2.1M	1.3M	0.8M	7.2M

Notes: The relative error of our method depends on the actual dynamic range, and for dynamic ranges 4.2, 5.9, 3.6, 4.8 the precisions in relative error are 0.015%, 0.021%, 0.013%, 0.017%, respectively.

4.2.4.2 Lossy Mode

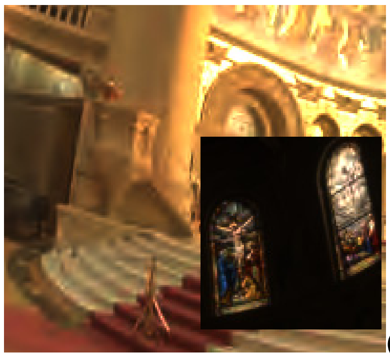
Our lossy compression scheme provides an efficient way to compress an HDR image in low bit rate and still keep high compressed image quality. In Table 4.4 we compare the results of our compression at various compression ratios with the compression result using Ward's and Mantiuk's compression schemes. The experimental data for Mantiuk's method are obtained from the compressed images kindly provided by its author. The data for Ward's method are obtained from the implementation of Ward's method kindly provided by Ward, and we ran it using the parameters provided in the demo made available by Ward ("-a 0.67 –b 0.75 –c full").

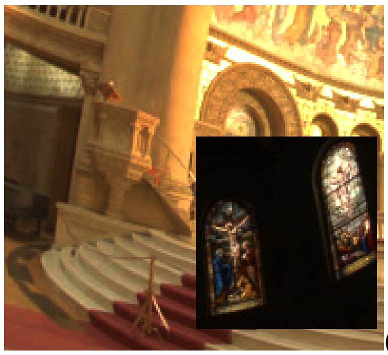
Table 4.4: Compression Statistics and Comparison with Ward's and Mantiuk's Methods

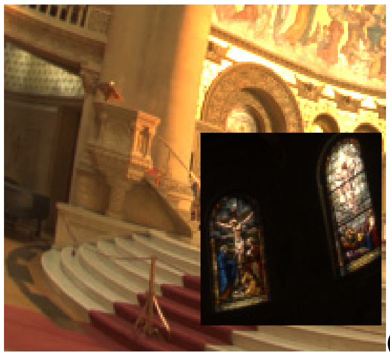
	Size(KB)	STDDEV	VDP(10^{-3})	Timing(s.)
rate=0.01	23	0.074	96	1.7/0.64
rate=0.05	118	0.022	34	1.7/0.68
rate=0.10	230	0.010	23	1.7/0.70
Ward's	125	0.040	46	1.1/0.58
Mantiuk's	138	0.032	44	N/A

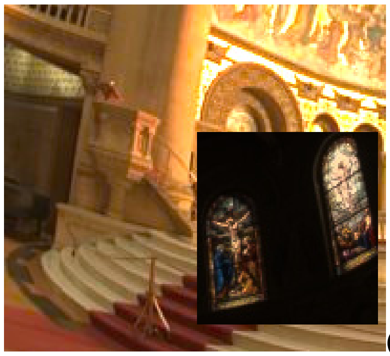
This 512 × 768 HDR image occupies 1.1MB in RGBE (RLE) and 823KB in OpenEXR (PIZ) format. The timing quoted in this table is from the runtime on an Intel Xeon 1.7G PC with 1Gbytes of memory running Windows XP. The comparison is caught out on the "memorial" image shown in Figure 4.12.

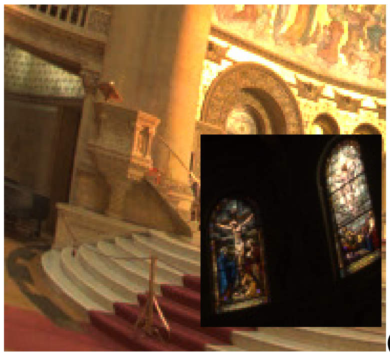
The results of compression are shown in Figure 4.11. "rate" is a parameter in JPEG2000, which specifies the ratio of desired data size to raw data size. The raw data size is the number of pixels times the number of bytes per pixel (e.g., 6 for n=16). STDDEV is the square root of the mean square error between the compressed image and the reference image in the logarithm domain. To have a metric that correlates with subjective perception, we make use of Lubin's visual difference predictor (VDP) [Lubin 1995], and use the mean value of the difference map as a visual fidelity indicator in this chapter. In the last column of Table 4.4 we show the compression and decompression times in seconds. It is seen that our algorithm consumes a considerable amount of time, but this is not a problem in encoding a single static HDR image.

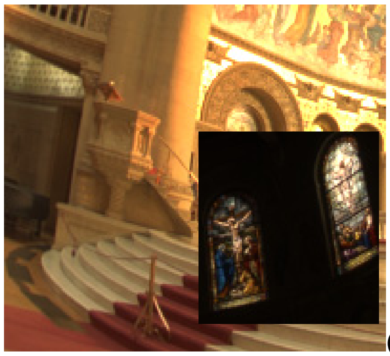

(a)


(b)

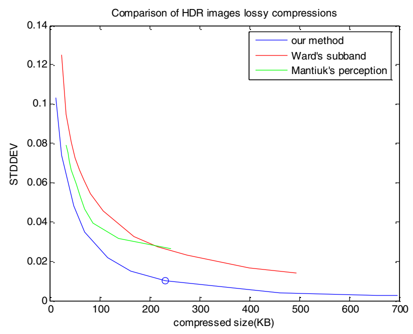

(c)


(d)


(e)


(f)

(g)



(h)

The “rate” for (a)-(c) are 0.01, 0.05, 0.10, respectively (23KB, 118KB, 230KB). (d) is compressed using Ward’s subband encoding (125K). (e) is compressed using Mantiuk’s perception based encoding (138KB). (f) is reference image. Inset is the darkened version of the rectangular area shown in the right. The relative positions of background images and the insets in (a)-(f) are shown as the blue boxes in (g). See Table 4.4 for the compression statistics.

Figure 4.12: Visual Quality Comparison of JPEG2000 Based HDR Image Data Compression

The compressed image in Figure 4.12 (a) shows some blur artifacts, but the compressed images in Figures 4.12(b) and (c) are indistinguishable from the reference image in Figure 4.12(f). In comparison, Figure 4.12(d) shows the compression result of Ward's subband encoding scheme [Ward 2004b], and Figure 4.12(e) uses Mantiuk's perception based encoding [Mantiuk 2004]. In this image we see visible artifacts in the brighter areas of the scene. The visual differences agree with the error predicted VDP. Thus, keeping the visual quality the same, our scheme produces compressed images at a bit-rate of about 1/5 of that achieved using Ward's subband encoding [2004b]. The last row of the table shows statistics of the compression result obtained with the lossy compression scheme of OpenEXR. Note that, though the error is much less, the poor compression rate (about 0.5) makes it completely uncompetitive. We also compare our lossy compression with Ward's subband encoding and Mantiuk's perception encoding, investigating the compression quality in terms of STDDEV changing with compressed size, as shown in Figure 4.12(h). Our method has obvious advantages, especially in small compressed sizes, i.e., low bit rate.

Figure 4.13 shows the compression results for three more HDR images. For the "park" HDR image in Figures 4.13(c), our HDR image encoder compresses the original HDR image (1,116K in RGBE format) to 118KB without introducing any visually distinguishable differences. In fact, in all the test HDR images in this figure (Figures (a), (c), (f)) compression rates greater than or equal to 0.05 produce results that are visually indistinguishable from the original HDR images (Figures (b), (d), (g)).

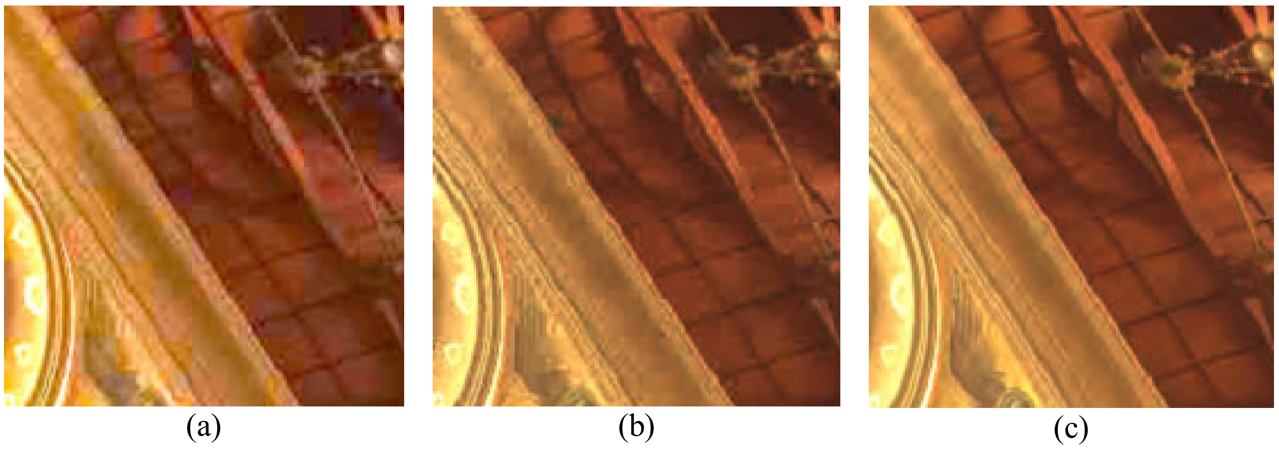

(a) Ward' subband method, compressed to 50.2KB, VDP: 79x10^{-3}; (b) Mantiuk' prerceptual method, compressed to 52.7KB, VDP: 73x10^{-3}; (c) Our method, compressed to 46.0KB, VDP: 63x10^{-3}

Figure 4.14: Comparison of Data Compression in Very Low Bit Rate

Figure 4.13(e) shows that our lossy compression performs quite well for very low bit rate and very high dynamic range. Figure 4.14 further illustrates our method converses the image quality even well in very low bit rate by comparing the image quality between our JPEG2000 based method (compressed to 46.0KB), Ward'd subband method (compressed 50.2KB), and Mantiuk's perceptual method (compressed to 52.7KB). For clarity, we show an inset from the top right part of the "memorial" image. The VDP confirms our method has the best quality when compressed to the same size.

4.2.5 Analysis

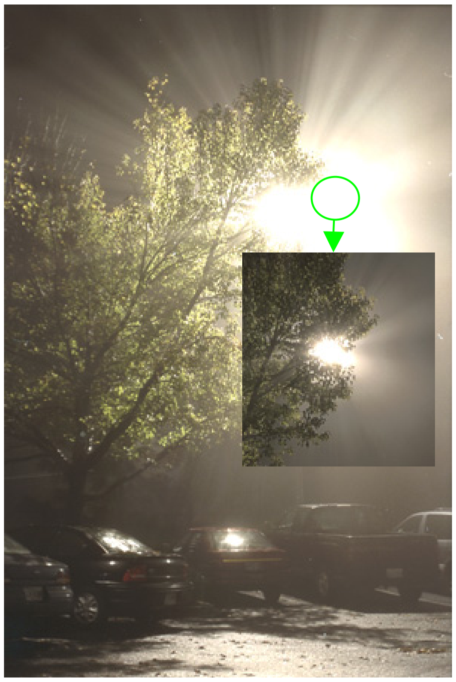
Our method extends an existing image compression technique (JPEG2000) to compress HDR images. It thus acquires other benefits from JPEG2000, like scalability, error resilience, and region of interest (ROI). Our wavelet-based approach is superior to any DCT based one; for example, it does not exhibits "blocking" artifacts in the wavelet based method. In contrast, this artifact issue is a serious problem for Mantiuk's and Ward's methods under low bit rates.

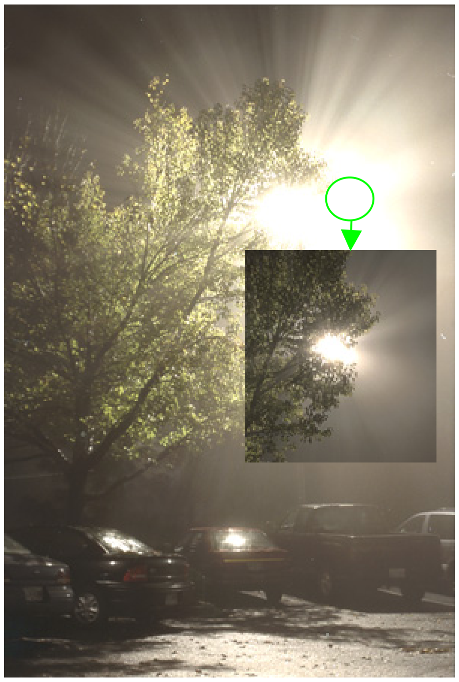
Compared to other lossy HDR compression schemes, our approach can reach the same visual quality at a much lower bit rate. It enables a minimum coding error (MSE) in the logarithm domain with any bit rate budget. For the “memorial” HDR image, even a 23K image is enough to achieve a visually good result.

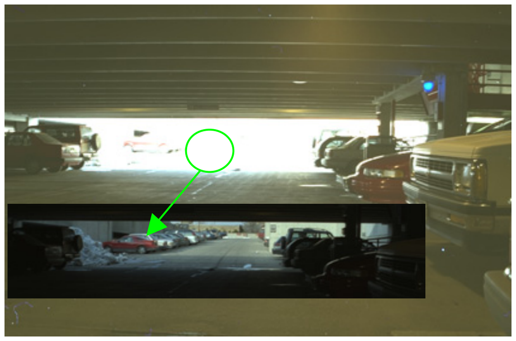
The quantization error using our approach is limited by the actual dynamic range and the maximum bit depth of the JPEG2000 implementation. The highest precision in the log domain using Jasper will be $R/(2^{16+1}-2)=R*0.00076\%$, where R is the actual dynamic range. For most natural HDR images, whose dynamic range covers up to 9 orders of magnitude, the pixel coding error in the logarithm domain is no more than 0.007%, with a corresponding relative error, according to Equation (4.16), of 0.03%, which is much less than 0.1%, the precision (relative error) of the half data type used in OpenEXR. The reason our pixel encoding can have higher precision than OpenEXR, while using the same number of bits, lies in the fact that we use the actual dynamic range, rather than nominal dynamic range of the half type.

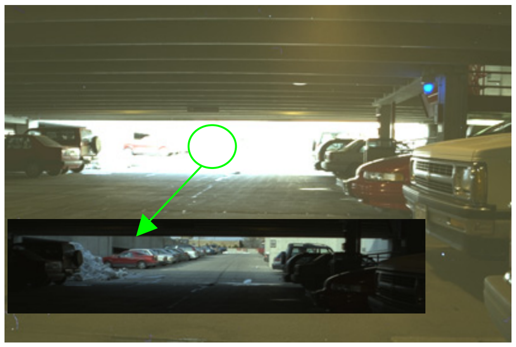
The lossless mode has a larger bit rate than OpenEXR, even than JPEG-LS on average [Rabbani 2002]. But the lossy mode is superior to others, particularly in low bit rates. Our approach provides a simple, straightforward, and efficient lossy HDR encoding.

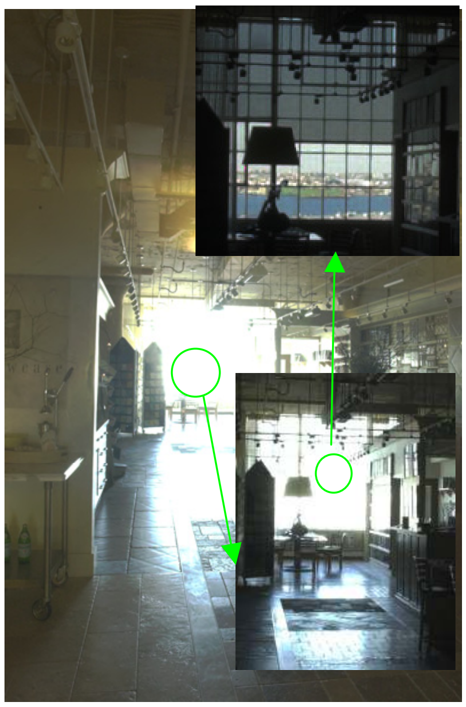
We would like the color transformation to be done before doing the log transformation. However, our attempt in doing so has brought about color artifacts in dynamic range areas. Though at the moment, we are not certain about the reason, we are tempted to believe that the issue could be in using the sRGB to YC_bC_r transformation which is designed mostly for low dynamic range images. As a part of our future work, we would like to find out the best uncorrelated color space and the appropriate transformation matrix.

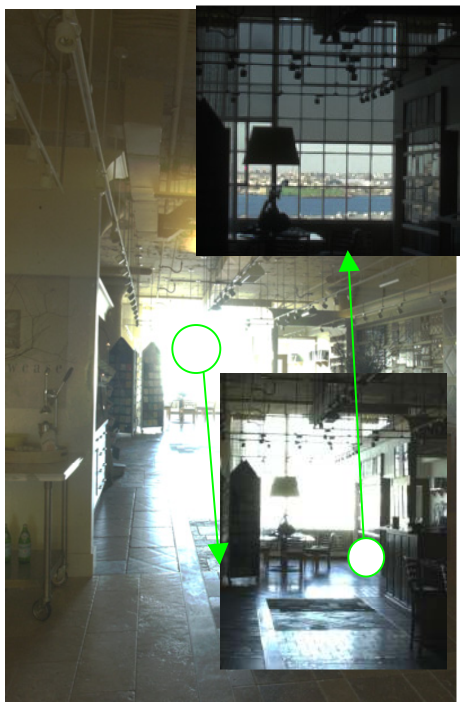

(a) BigFogMap, rate=0.05 (248K)

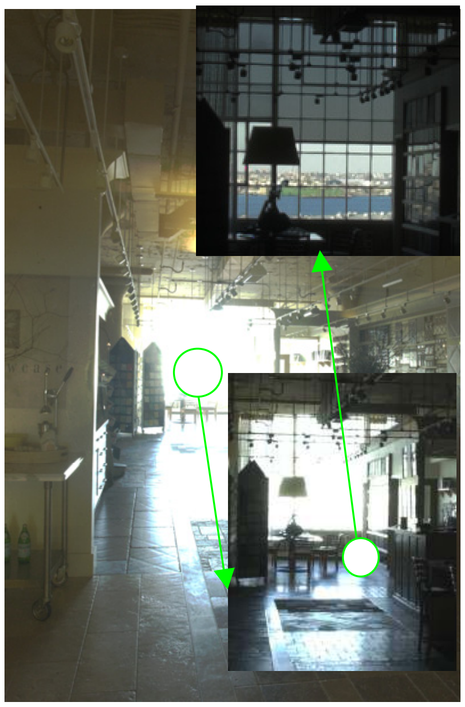

(b) BigFogMap, reference (2,683K RGBE RLE)


(c) Park, rate=0.05 (108K)


(d) Park, Reference (1,142K RGBE RLE)


(e) designcenter 0.01 (154K)


(f) designcenter 0.05 (766K)


(g) designcenter reference (8,356K RGBE RLE)

(a), (c), (e) and (f) are compressed images of BigFogMap, Park, and designcenter, respectively. (b), (d) and (g) are corresponding reference images. (a), (b) are courtesy of Tumblin. (c), (d) are courtesy of Pattanaik, and (e)-(g) are courtesy of Durand.

Figure 4.13: Lossy JPEG2000 Based HDR Image Compression Results

One simple improvement we can make involves optimizing the logarithmic operation by taking advantage of the format definition of floating point numbers whose codes include a mantissa and an exponent. This remains to be done.

The choice of view independent quantization is deliberate in our HDR image encoding scheme. The reason is: the human eyes are not at a fixed adaptation level when viewing HDR images, and hence will warrant an adaptive quantization for the wavelet coefficients as a function of pixel position. Though it is not impossible to address this, such consideration requires careful research that can build on previous work concerning adaptive quantization of conventional images [Strutz 2001; Nadenau 1999]. The human eye is more sensitive to luminance than chrominance, which is exploited by the JPEG standard by subsampling the chrominance channel. It is also possible to encompass this property in the HDR image encoding for further optimization. We leave incorporation of visual perception into our compression scheme as a topic of future research.

It is possible to extend our approach to compress HDR video based on MJPEG2000 (see Part 3 of [ISO/IEC 2000]). HDR video can be compressed simply by sending each single frame to our HDR image compressor. However, the decoding time of JPEG2000 is rather slow (0.5 sec for a 512 by 768 HDR image). The GPU implementation of JPEG2000 [Wang 2004] is much faster, and it may be used for a real-time HDR video codec.

CHAPTER FIVE: MONTE CARLO NOISE REDUCTION

We have discussed in the introduction chapter that Monte Carlo noise is an inescapable artifact in synthetic images rendered using Monte Carlo methods in low sampling rate, and it is necessary to remove the noise to have a high quality synthetic image. This chapter presents two novel post-processing based methods for Monte Carlo noise reduction in synthetic images. The first one finds the noise distribution in the wavelet domain, and applies a Bayesian method to suppress the noise [Xu 2005b]. The other applies bilateral filtering to suppress the outlier and the incoherence in a unified manner.

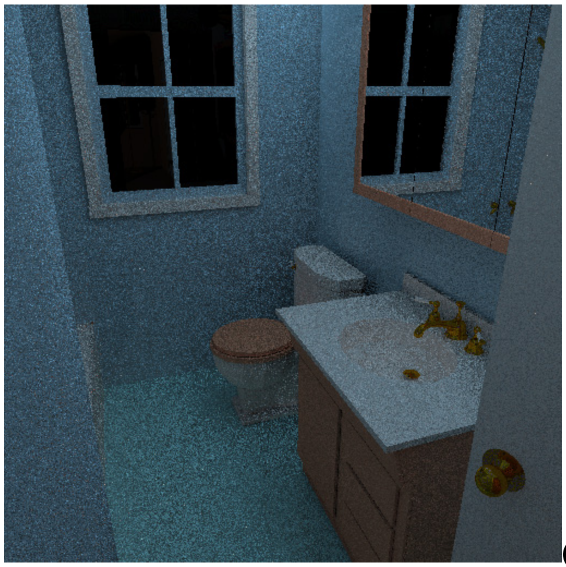
Monte Carlo noise comes from the variance due to limited sampling rate in Monte Carlo integration of rendering equation, and exhibit as either "outliers" or "inter-pixel incoherence". "outliers" are the Monte Carlo noise that exhibits as standalone bright pixels; and "inter-pixel incoherence" is the Monte Carlo noise that exhibits as small but visible discontinuities between neighboring pixels inside smooth regions.

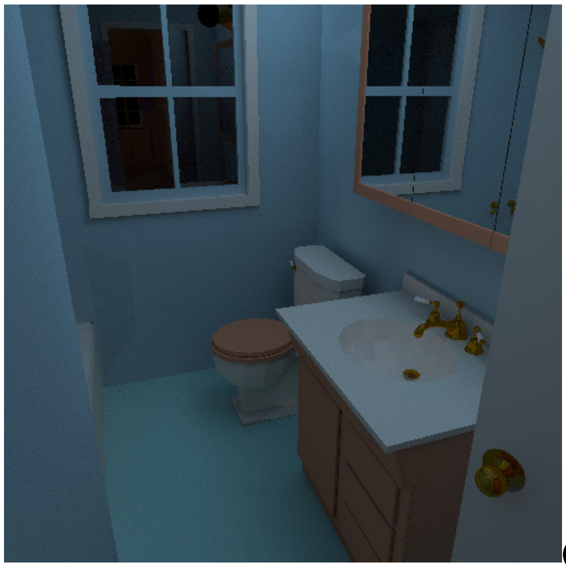
The methods proposed in this chapter can effectively suppress the noise. Thus, it is possible to generate fast high quality rendering efficiently using Monte Carlo based methods.

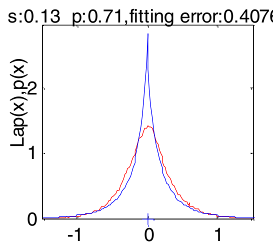
5.1 Bayesian Based Noise Reduction

This section first presents our findings on the statistical characteristics of the Monte Carlo noise, and then proposes a Bayesian method to remove this noise. The aim of this approach is to efficiently produce high quality synthetic images using Monte Carlo based rendering at low sampling rates.

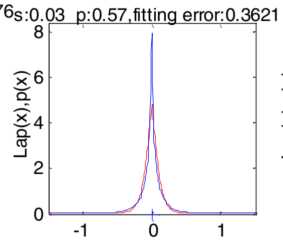
This work has two contributions: (1) it proposes a general model of the Monte Carlo noise; (2) it is the first attempt of applying Bayesian method to Monte Carlo noise reduction.


(a)

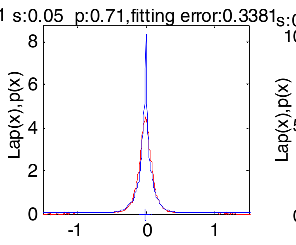

(b)



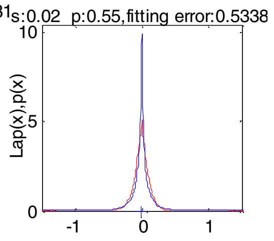
(c)



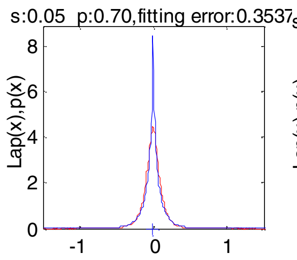
(d)



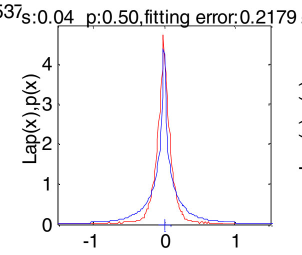
(e)



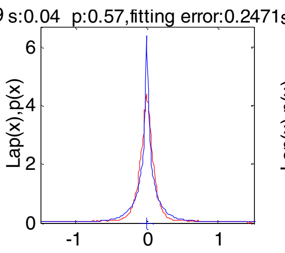
(f)



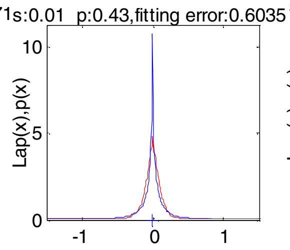
(g)



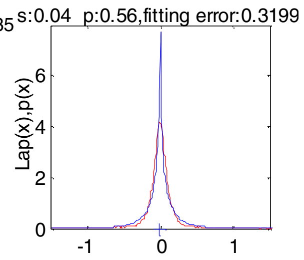
(h)



(i)



(j)



(k)

Figure (c)-(k) are distribution functions for high pass band, and for bands (1,1), (l,2), (1,3), (1,4), (2,1), (2,2), (2,3), (2,4), respectively, where (x,y) means "level x, band y".

Figure 5.1: Distribution of Monte Carlo Noise

5.1.1 Monte Carlo Noise Modeling

The Monte Carlo noise contaminating the diffuse inter-reflection component of rendered image is modeled in wavelet domain using a generalized Laplacian distribution. Given an image created in a short time using a Monte Carlo method, there is plenty of visible noise. These noises are generally present in two forms: outliers and inter-pixel incoherence. We try to build a general statistical model to handle both types of Monte Carlo noise. This model addresses two issues: the way noise is combined with true pixel values; and the distribution of the noise.

From our experiment with a large collection of images generated using Monte Carlo methods, we find that Monte Carlo noise is most likely multiplicative in nature, and the coefficients of the wavelet band of the Monte Carlo noise map in log domain approximately follow a regular distribution. We use parameterized Laplacian shown in Equation (5.1) to model the distribution of these noise coefficients in wavelet domain.

$$\begin{aligned} f_{MCnoise}(x; s, p) &= \frac{1}{Z} e^{-|x/s|^p}, -\infty < x < \infty \\ Z &= 2\frac{s}{p}\Gamma(\frac{1}{p}) \end{aligned} \tag{5.1}$$

where, s, p are parameters of the distributions, and Z is normalization constant. s specifies the heaviness of the noise, and p specifies the shape of the distribution function.

The Laplacian function has been used in [Mallat 1989] to model the distribution of wavelet coefficients of natural images.

To verify the correctness of our model, we give examples as shown in Figure 5.1. We use the measure shown in Equation (5.2) to estimate the fitting error.

$$\mathit{fitting\ \ error} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(p(x) - f(x))^2 / f(x)^2} \qquad (5.2)$$

where, $p(x)$ is the true distribution density of noise at point x, $f(x)$ is the modeled distribution density and N the number of bins used in the discrete summation. The smaller is the *fitting error*, the greater is the fitting accuracy. The two images Figure 5.1(a) and Figure 5.1(b) are rendered using the *Radiance* software [Ward 1998]. In Figure 5.1(a) indirect reflection component is estimated using 10 samples per bounce, and Figure 5.1(b) is estimated using 300 samples per bounce. Figure 5.1(a) takes 0.0794 hours on a Celeron 2.0G running Windows 2000, while figure 5.1(b) takes 1.6653 hours on the same platform. So, the computation for higher samples also takes too much time. Figure 5.1(b) is used as accurate image. Noise map is extracted by dividing Figure 5.1(a) with Figure 5.1(b). The noise map is translated into logarithm domain and converted into wavelet domain using the steerable filters [Simoncelli 1996; Simoncelli 1999]. We compute the high pass band, as well as 4 bands for 2 levels. The coefficient distribution and the fitting to Laplacian function for these 9 bands are shown in Figure 5.1(c) through Figure 5.1(k). Figure 5.1(c) is for high pass band, Figure 5.1(d)-(g) are for the 4 bands in level 1, and Figure 5.1(h)-5.1(k) are for the 4 band in level 2. The blue curves are the fitted Laplacian functions, and the red curves are the actual distribution density. The title line on the figures show the s, p values and the error in fitting the Laplacian function computed using Equation 5.2.

The parameters of fitting Laplacian function for the bands in Figure 5.1 are shown in Table 5.1.

The results from various experiments show that for most scenes:

- p often lies in the range $[0.5, 1.5]$, and the p values for all bands are generally similar.
- s often lies in the range $[0.0, 1.0]$, and the s values for bands except high pass band are generally similar, and is usually one quarter to one half of the value for high pass band.

Based on these two observations we use two parameters, (s_n, p_n) for all bands except for high pass band, and use $(2s_n, p_n)$ for high pass band. So, only two parameters are used to model the noise, (s_n, p_n). For heavier noise, we use a larger s in $[0.0, 0.15]$, and for more complex scenes we use a smaller p in $[0.5, 1.0]$. We would like to point out that the above selection rules are based solely on our experimental observation. We are yet to find a theoretical justification.

Table 5.1: Fitting Laplacian parameters for noise in images in Figure 5.1

	s	p	fitting error
High pass band	0.1345	0.7111	0.4076
Level 1, band 1	0.0288	0.5658	0.3621
Lebel 1, band 2	0.0479	0.7140	0.3381
Level 1, band 3	0.0212	0.5478	0.5338
Level 1, band 4	0.0454	0.6998	0.3537
Level 2, band 1	0.0385	0.5015	02179
Level 2, band 2	0.0369	0.5654	0.2471
Level 2, band 3	0.0114	0.4327	0.6035
Level 2, band 4	0.0370	0.5648	0.3199

5.1.2 Bayesian Monte Carlo Noise Reduction

5.1.2.1 Denoising Framework

Based on the Monte Carlo noise model given in the previous section, a general Bayesian denoising framework is described in this subsection. The framework has been shown in Figure

5.2. Following [Jensen 1995] we assume that most Monte Carlo noise comes from indirect inter-reflection. So, we first separate the rendering result into direct component (direct illumination + specular illumination) and indirect component (diffuse inter-reflection), which is easy to implement by adding a few lines of code into the renderer source code to separately record the indirect and direct components.

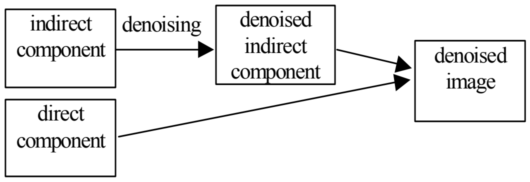
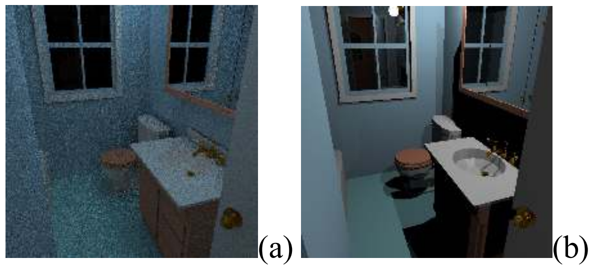


Figure 5.2: Bayesian Monte Carlo Denoising Framework

Indirect component is denoised, and then combined with direct component to generate the final denoised image. Figure 5.3 shows the direct and indirect components of a rendered image. The next subsection presents the Bayesian denoising method, which makes use of Monte Carlo noise model shown in this section.



(a) is Indirect component, and (b) is Direct component.

Figure 5.3: Decomposition of Synthetic Image into Direct and Indirect Components

5.1.2.2 Bayesian Denoising

We apply Bayesian denoising in the wavelet domain [Simoncelli 1996; Simoncelli 1999] to estimate the true image value from noisy value. The image is first transformed into logarithm domain, and then transformed into wavelet domain. Bayesian method is then applied to remove image noise by adjusting the transformed wavelet coefficients. The method is based on the assumption that the noise is independent of the true value. According to the observation result from previous sub-section, the noisy image value can be written as multiplication of true value and noise value.

$$Y=C*N \tag{5.3}$$

where, Y is noisy image value, C is true image value, and N is noise value.

When taking the logarithm and then wavelet transformation on Equation (5.3), the right side becomes addition of log true band coefficient and log noise band coefficient. We use lower case symbols y, c, n to denote the wavelet band coefficients of $\log Y$, $\log C$ and $\log N$, respectively. Thus, we get Equation (5.4).

$$y=c+n \tag{5.4}$$

As stated in [Mallat 1989; Simoncelli 1996; Simoncelli 1999], wavelet band coefficients of the natural images follow Laplacian distribution. Inspired by this observation result, we further find that the band coefficients of logarithms of natural images also follow Laplacian distribution. Thus we model the distribution of wavelet band coefficients, c, using Equation (5.5).

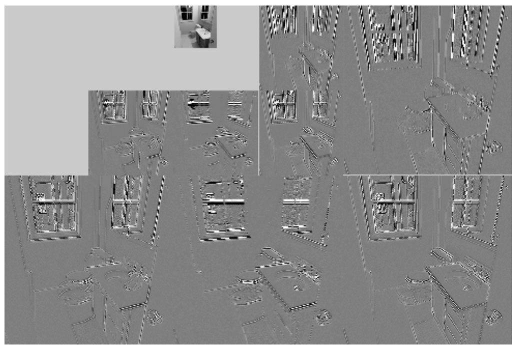
$$\begin{gathered} P_c(c; s_c, p_c) = \frac{1}{Z} e^{-|c/s_c|^{p_c}} \\ Z_c = 2\frac{s_c}{p_c}\Gamma\left(\frac{1}{p_c}\right) \end{gathered} \tag{5.5}$$

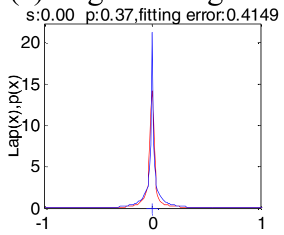
where, s_c, p_c are parameters of the distributions, and z_c is the normalization constant.

Table 5.2: *s, p* and *fitting error* for Image in Figure 5.1

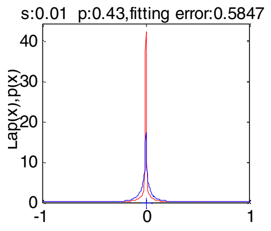
	s	p	fitting error
High pass band	0.0033	0.3732	0.4149
Level 1, band 1	0.0060	0.4329	0.5847
Lebel 1, band 2	0.0059	0.4654	0.4269
Level 1, band 3	0.0022	0.3833	0.4673
Level 1, band 4	0.0036	0.4294	0.3991
Level 2, band 1	0.0416	0.5293	0.7867
Level 2, band 2	0.0387	0.5916	0.7241
Level 2, band 3	0.0096	0.4294	0.6356
Level 2, band 4	0.0246	0.5369	0.6610

(a) original image


(b) wavelet transformation



(c)Distribution of high pass band [s, p]=[.003, .373]



(d)Distribution of band 1 level 1 [s, p] =[.006, .433]

Figure 5.4: Test Image, Wavelet Transformation and Distributions

Figure 5.4 shows the distribution of wavelet band coefficients of our test image and Laplacian fit for the distribution.

The wavelet used is a steerable wavelet [Simoncelli 1999], Red lines in the figures represent the actual distribution, and the blue ones represent the fitting curves. For more details and the source code of steerable pyramid, refer to the link [Simoncelli 2004].

Following the sub-band coefficient estimation method proposed in [Simoncelli 1999], the true sub-band coefficient is estimated by Equation (5.6).

$$\hat{c}(y) = \int p_{c|y}(c \mid y) c\,dc \tag{5.6}$$

where, $p_{c|y}$ $(c|y)$, the posterior probability, is the probability of actual coefficient as c given the observed coefficient y. Using Bayesian rule, it is possible to express $p_{c|y}$ $(c|y)$ in terms of the components that are known in advance.

$$\begin{aligned} p_{c|y}(c \mid y) &= \frac{p_{y|c}(y \mid c) p_c(c)}{p_y(y)} \\ &= \frac{p_{y|c}(y \mid c) p_c(c)}{\int p_{y|c}(y \mid c) p_c(c) dc} \\ &= \frac{p_{y-c|c}(y-c \mid c) p_c(c)}{\int p_{y-c|c}(y-c \mid c) p_c(c) dc} \\ &\overset{n=y-c}{=} \frac{p_{n|c}(y-c \mid c) p_c(c)}{\int p_{n|c}(y-c \mid c) p_c(c) dc} \end{aligned} \tag{5.7}$$

In Equation (5.7), $p_{y|c}(y \mid c)$ is the posterior probability of y given c. $p_c(c), p_y(y)$ are a prior probability of c and y. $p_{y-c|c}(y-c \mid c)$ is the posterior probability of noise $(y\text{-}c)$ given c. Because the Monte Carlo noise is assumed to be additive in logarithm domain, and independent of the true band coefficient value c the posterior probability of noise n given c, is simply the probability of the noise itself, *i.e.*

$$p_{n|c}(y-c \mid c) = p_n(y-c) \tag{5.8}$$

Equation (5.7) can now be rewritten as Equation (5.9) by plugging in Equation (5.8).

$$p_{c|y}(c|y) = \frac{p_n(y-c)p_c(c)}{\int p_n(y-c)p_c(c)dc} \quad 5.9)$$

Using Equation (5.9), the estimator of true band wavelet coefficient in Equation (5.6) can be written as Equation (5.10) [Simoncelli 1999].

$$\hat{c}(y) = \frac{\int p_n(y-c)p_c(c)cdc}{\int p_n(y-c)p_c(c)dc} \quad (5.10)$$

$p_n(y-c)$, the distributions of the noise and $p_c(c)$, the true wavelet sub-band coefficients are both modeled as Laplacian distributions, as shown in Monte Carlo noise analysis in last section and image wavelet sub-band coefficients analysis in this section. Given the values of these distributions, Equation (5.10) can be calculated using simple discrete integration method.

Both of the distributions are determined by two parameters *s* and *p*. For the noise, the parameters (s_n, p_n) are provided as input to the denoising program. And, for the image wavelet sub-band coefficients, the parameters are recovered from second and fourth moments of noisy wavelet band coefficients by solving the Equation (5.12). (For details about the derivation of Equations (5.11), refer to Appendix B.) In the Equation (5.11), (s_n, p_n) are parameters of the noise distribution, and (s_c, p_c) are parameters for the accurate sub-band coefficients, σ_y^2, m_y^4 are variance and fourth moments of the noisy sub-band coefficients respectively, and Γ is the gamma function. (s_n, p_n) are provided as program parameters. Since there are two unknowns, s_c, p_c in two equations, the parameters are uniquely determined from the equation pair.

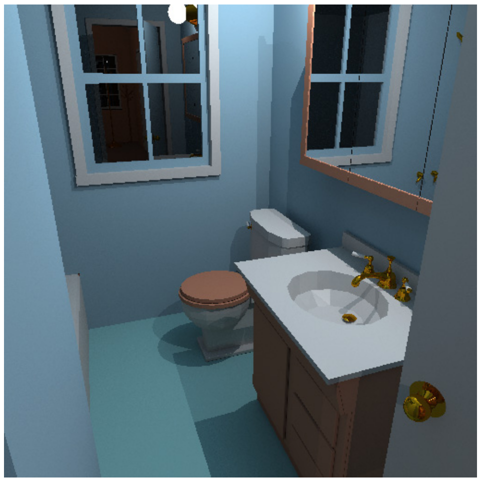
$$
\begin{aligned}
\sigma_y^2 &= \frac{s_c^2\Gamma(3/p_c)}{\Gamma(1/p_c)} + \frac{s_n^2\Gamma(3/p_n)}{\Gamma(1/p_n)} \\
m_y^4 &= \begin{pmatrix} \dfrac{s_c^4\Gamma(5/p_c)}{\Gamma(1/p_c)} + \dfrac{s_n^2\Gamma(5/p_n)}{\Gamma(1/p_n)} \\ + \dfrac{6s_c^2 s_n^2\Gamma(3/p_c)\Gamma(3/p_n)}{\Gamma(1/p_c)\Gamma(1/p_n)} \end{pmatrix}
\end{aligned}
\tag{5.11}
$$

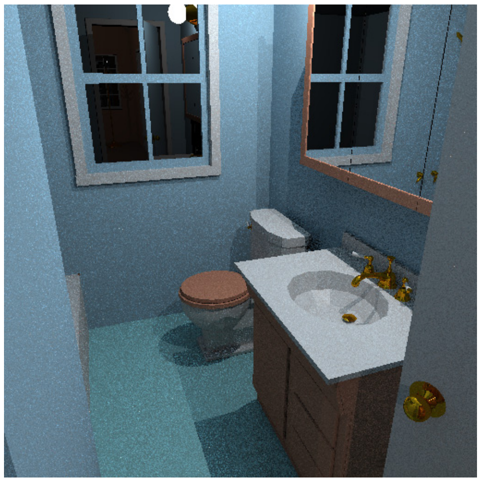
With the recovered parameters of the wavelet sub-band coefficients distribution, the estimator of accurate sub-band coefficient can be easily computed through Equation (5.10). The integral is computed using discrete summation of integrand. Finally, the denoised image is recovered by transforming the denoised wavelet sub-band coefficients into spatial domain.

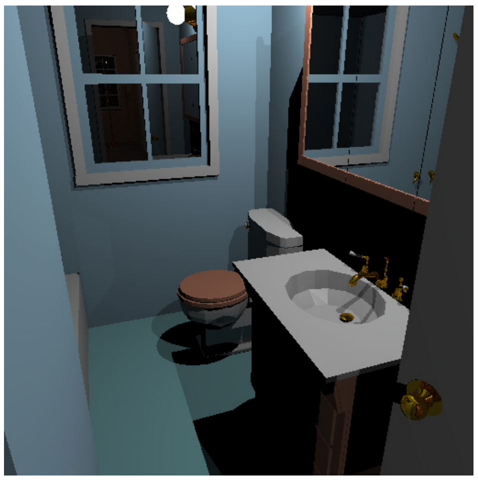
5.1.3 Experimental Results

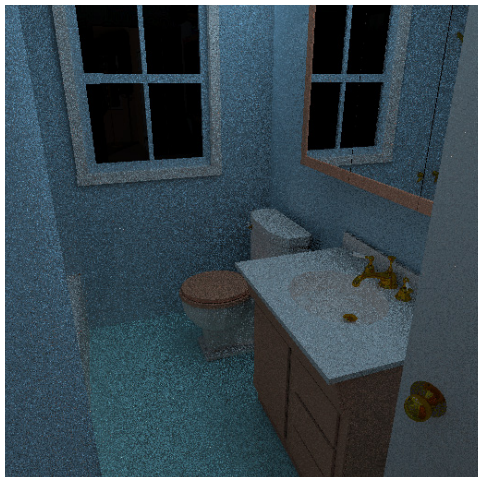
To show the denoising effects of our approach, two denoised images using our method are shown in this sub-section. We used the *Radiance* software [Ward 1998] to create the noisy and accurate images. For the "cabin" image, as shown in Figure 5.5, computation of the noisy image took 285.84 secs. The denoising took 36.61 secs. Thus the total time spent was 322.45 secs, compared to 3601.67 secs taken to compute an equivalent image. The denoising was carried out on a 2.0G Celeron running Windows 2000. We implemented our approach using Matlab 6.0.

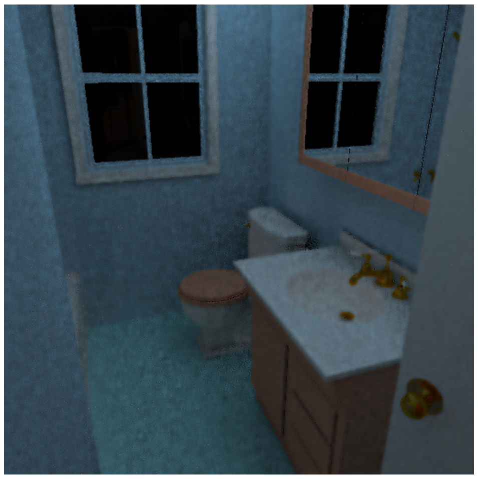
The noisy office image in Figure 5.5 is composed of direct component (c) and indirect component (d), which are rendered using 10 indirect samples. Accurate image using 300 samples is shown in Figure 5.5(a), noisy image is shown in Figure 5.5(b) for comparison with denoised results. The denoised indirect component is shown in Figure 5.5(e), and the final denoised image is shown in Figure 5.5(f). The noise parameters used in denoising Figure 5.5(d) is (s_n=0.19, p_n=1.5). Note that the direct component in Figure 5.5(c) carries little noise.


(a) Accurate image


(b) Noisy image


(c) Direct component


(d) Noisy indirect image


(e) Denoised indirect image

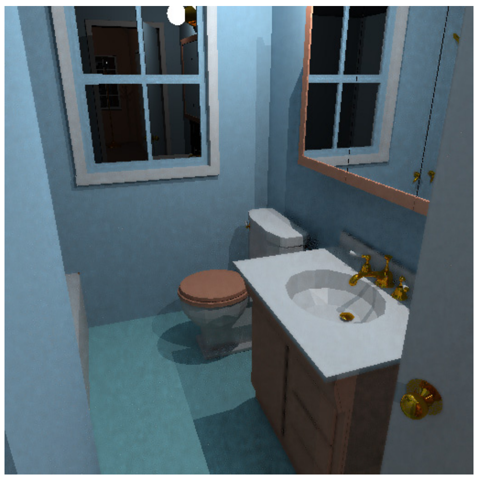

(f) Denoised image

Figure 5.5: Bayesian Denoising Results of "office"

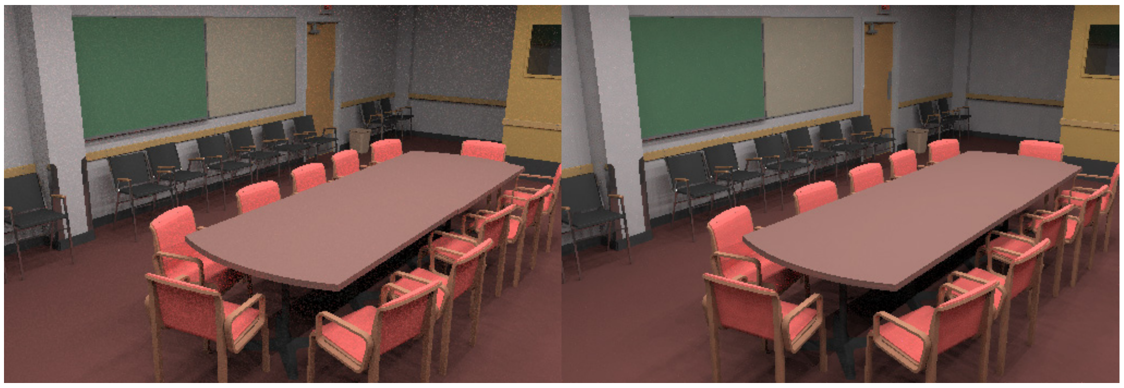

(a) Noisy image "conf" (b) Denoised image "conf"

Figure 5.6: More Bayesian Denoising Examples

Figure 5.6 illustrates another denoising example. The used parameters are s_n=0.6, p_n=1.5.

We can see from these experimental results that the edges are well preserved, and the noise is suppressed with little blurring the image.

The experimental results also verify our findings about Monte Carlo noise. Based on the assumption that most noises gives rise to smaller wavelet coefficients Bayesian denoising method works by suppressing smaller band coefficients, but keeping larger band coefficients. Its successful application to Monte Carlo noise verifies that most Monte Carlo noise is actually concentrated around smaller values. Our Laplacian modeling of Monte Carlo noise also has the greatest density around the smallest values.

5.1.4 Analysis of Bayesian Monte Carlo Noise Reduction

We have presented a general framework of the Monte Carlo noise removal in this section. Based on our observation, we have presented a novel model of Monte Carlo noise. Bayesian method effectively exploits this model for noise reduction. Nice looking images can be synthesized using a combination of low sample rendering and the noise removal technique proposed in this research.

Compared to the other Monte Carlo denoising method [Jensen 1995; Rushmeier 1994; McCool 1999], we take a statistical perspective to the Monte Carlo noise, and introduce Bayesian method to remove Monte Carlo noise under this perspective. This is the first trial to reduce Monte Carlo noise by modeling its statistical characteristics. Our experimental results prove its feasibility, and we believe more work can be done along this new direction for Monte Carlo noise reduction problem.

So far, we have only applied our technique on the most commonly encountered Monte Carlo noise, that is, Monte Carlo noise generated using a path tracer. We have not tried on other special Monte Carlo noise, like Metropolis noise [Veach 1997], which is presented as streaks. Fortunately, those special Monte Carlo noises are less frequent present in synthetic images generated by Monte Carlo path tracing methods.

However, proof of the assumption in our approach remains another problem yet to be studied. We derive our approach by assuming the independence of Monte Carlo noise and its contaminated true value. Our experiments show successful results under this assumption. Because the emphasis of this section is to present the successful application of Bayesian method on Monte Carlo noise reduction with a noise statistics model, we leave this problem for later study.

5.2 Bilateral Filtering Noise Reduction

Another novel Monte Carlo noise reduction operator is proposed in this section. We apply and extend the standard bilateral filtering method and build a new local adaptive noise reduction kernel. It first computes an initial estimate for the value of each pixel, and then applies bilateral filtering using this initial estimate in its range filter kernel. It is simple both in formulation and implementation. The new operator is robust and fast in the sense that it can suppress the outliers, as well as the inter-pixel incoherence in a non-iterative way. It can be easily integrated into existing rendering systems as a post-processing step. The results of our approach are compared with those of other methods. A GPU implementation of our algorithm runs in 500ms for a 512×512 image.

Our work is inspired by the work of Tomasi et al. [1998], where bilateral filtering is proposed to smooth images while keeping the edges undisturbed. Bilateral filtering is also successfully applied to image denoising [Elad 2002a; Elad 2002b], mesh smoothing and denoising [Jones 2003; Fleishman 2003], and high dynamic range tone mapping [Durand 2002]. A theoretical analysis of this technique is presented in [Barash 2001]. The principle of bilateral filtering is simple. It combines the domain filtering and range filtering, as shown in Equation (5.12).

$$h(x) = \frac{\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(\xi)c(\xi,x)s(f(\xi),f(x))d\xi}{\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} c(\xi,x)s(f(\xi),f(x))d\xi} \tag{5.12}$$

where, $h(x)$ is the estimator of the current pixel x, $f(x)$ is the pixel value of x and $f(\xi)$ is the pixel value of its neighbor(s) ξ, and $s(f(\xi),f(x))$ and $c(\xi,x)$ are the range filter and domain filter kernels. They are often modeled as Gaussian functions with parameters σ_r,σ_d respectively, as shown in Equation (5.13).

$$\begin{aligned} c(\xi,x) &= \exp\left\{-\frac{1}{2}\left(\frac{|\xi - x|}{\sigma_d}\right)^2\right\} \\ s(f(\xi),f(x)) &= \exp\left\{-\frac{1}{2}\left(\frac{|f(\xi) - f(x)|}{\sigma_r}\right)^2\right\} \end{aligned} \tag{5.13}$$

If some neighbor ξ is an outlier, it has a much larger or much smaller value $f(\xi)$ than that of the central point x. Its contribution to the estimator $h(x)$ will be greatly reduced by the range filter $s(f(\xi),f(x))$ which favors similar range values rather than disparate values. Bilateral filter is a robust local adaptive filter, which can be used to enhance image coherence. However, as illustrated in Figure 5.7(b) and (c), it cannot be directly used to suppress the outliers of Monte Carlo noise. In the next section, we show that the original bilateral filtering is not as robust as claimed in [Durand 2002; Jones 2003]. We extend the standard bilateral filtering to handle outliers and inter-pixel incoherence in a unified framework. The contributions are:

- Application of bilateral filtering to Monte Carlo noise reduction.
- Extension of bilateral filtering with an initial estimation preprocess.

The rest of this section is organized as follows. Sub-section 5.2.1 presents our Monte Carlo noise operator developed from bilateral filter. Sub-section 5.2.2 describes a denoising framework, which can be easily integrated into existing rendering system. Experimental results and analysis are given in the last two sub-sections.

5.2.1 Monte Carlo Noise Reduction Operator

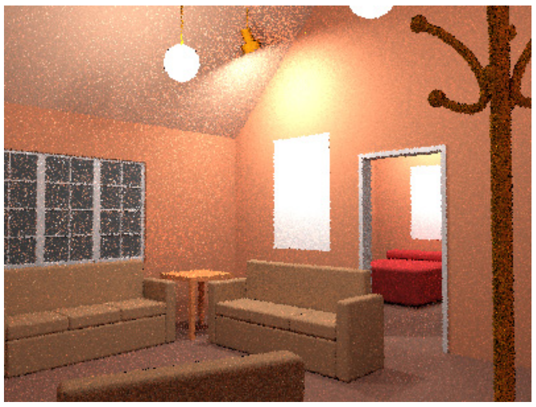
The outliers in Monte Carlo noise are pixels with much larger or much smaller values than its neighbors. It is desirable to remove them together with inter-pixel incoherence while keeping edges undisturbed. A Gaussian filter blurs the edges. Therefore, McCool et al [1999] introduced anisotropic diffusion to suppress inter-pixel incoherence while keeping edges intact. Standard bilateral filtering can do the same thing as anisotropic diffusion, but neither of them can effectively remove the outliers. This is because the initial estimator $f(x)$ used in $s(f(\xi),f(x))$ is far different from its true value, and very little contribution to $h(x)$ comes from such neighbors due to the infinitesimal weights returned by the range function. Thus, the outliers remain almost unchanged. They are neither suppressed, nor do they contribute to their neighbors. As shown in Figures 5.7(b) and 5.7(c), the outliers remain there after applying standard bilateral filtering. Fortunately, standard bilateral filtering can be extended to suppress both outliers and inter-pixel incoherence while keeping edges intact. We propose to employ an initial near-true estimator $\tilde{f}(x)$ to replace $f(x)$, and make use of Equation 5.14 as our new Monte Carlo noise reduction operator. Note that we use $\hat{f}(x)$ to replace $h(x)$ to denote the new estimator using bilateral filtering around point x.

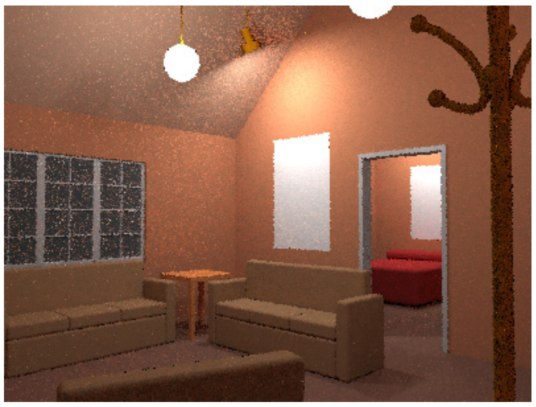
$$\hat{f}(x) = \frac{\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(\xi)c(\xi,x)s(f(\xi),\tilde{f}(x))d\xi}{\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} c(\xi,x)s(f(\xi),\tilde{f}(x))d\xi} \tag{5.14}$$

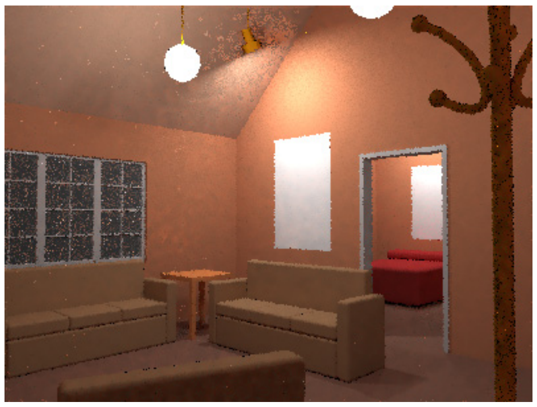
There are various possible options for $\tilde{f}(x)$, such as mean value around pixel x, or median value around pixel x. From our experiments, we find that Gaussian filtered value (shown in Equation 5.15) performs the best in dealing with Monte Carlo noise.

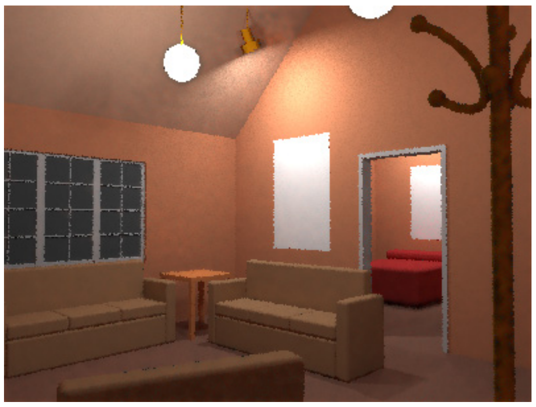
$$\tilde{f}(x) = \frac{\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(\xi)c(\xi,x)d\xi}{\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} c(\xi,x)d\xi} \tag{5.15}$$

Figure 5.7 shows the denoising of “living room” (see http://radsite.lbl.gov/) using original bilateral filtering, iterative bilateral filtering, and our bilateral filtering extension. The Gaussian parameters used in all the cases are the same: $\sigma_r = 2.0, \sigma_d = 0.4$. Standard bilateral filtering (Figure 5.7(b)) is almost ineffective in reducing Monte Carlo noise. In Figure 5.7(c), the bilateral filtering is iterated 20 times [Elad 2002a], and the incoherence inside regions are well suppressed, but the outliers remains unchanged. Notice the outliers on the window of Figure 5.7(c). It shows that the standard bilateral filtering is not so robust in suppressing outliers!


(a)


(b)


(c)


(d)

(a) Noisy image; (b) Standard bilateral filtering; (c) Iterative bilateral filtering used in [Elad 2002a] 20 iterations. (d) Our new bilateral filtering operator.

Figure 5.7: Outliers Reduction using Bilateral Filtering

Figure 5.7(d) shows the success of our extension to bilateral filtering. Our Monte Carlo noise reduction operator can also be used to reduce noises other than Monte Carlo noise.

5.2.2 Numerical formulation

The integration in Equations (5.16) and (5.17) are evaluated as discrete summation. As the weight function is very small at a distance farther than $3\sigma_d$ away from the central pixel ($e^{-(3\sigma_d)^2/(2\sigma_d^2)} = e^{-9/2} < 0.012$), we select a square window around the current pixel of size $6\sigma_d \times 6\sigma_d$ as the neighborhood window. The discrete version of the equations in this window is shown in Equations (5.16) and (5.17).

$$\hat{f}(i,j) = \frac{\sum_{u=-3\sigma_d}^{3\sigma_d} \sum_{v=-3\sigma_d}^{3\sigma_d} f(i+u, j+v) c(u,v) s(f(i+u, j+v), \tilde{f}(i,j))}{\sum_{u=-3\sigma_d}^{3\sigma_d} \sum_{v=-3\sigma_d}^{3\sigma_d} c(u,v) s(f(i+u, j+v), \tilde{f}(i,j))} \tag{5.16}$$

where,

$$\tilde{f}(i,j) = \frac{\sum_{u=-3\sigma_d}^{3\sigma_d} \sum_{v=-3\sigma_d}^{3\sigma_d} f(i+u, j+v)c(u,v)}{\sum_{u=-3\sigma_d}^{3\sigma_d} \sum_{v=-3\sigma_d}^{3\sigma_d} c(u,v)} \tag{5.17}$$

Our computation first finds the initial estimated value $\tilde{f}(i,j)$ for each pixel. Then, a bilateral filtering step is executed using $\tilde{f}(i,j)$. It is a non-iterative process, and the computation is fast. The denoising effects are greatly enhanced by this single additional initial estimation step, as shown in Figure 5.7(d). The pseudocode is briefly described in Table 5.3. The whole process is a loop over each pixel p in the image. Inside the loop first the range filter kernel is constructed using parameter σ_r and initial estimator $\tilde{f}$ and then is combined with the domain filter kernel. The resulting bilateral filter kernel is centered on p and integrated with the pixels within the square window of size $6\sigma_d \times 6\sigma_d$ to get the filtered value for the pixel *p*.

5.2.3 Denoising Framework Using Bilateral Filtering

As described by Jensen [Jensen 1995], most of the noise arises from computing diffuse inter-reflection (indirect component) using Monte Carlo methods. The contribution from direct illumination and specular inter-reflection (direct component) carries little noise. We follow this observation, and denoise only the indirect component. The denoised indirect component is then added to the direct component to get the final denoising result. The direct and indirect components are easily separated by adding only a few lines of code into the Monte Carlo renderer. The whole denoising process is briefly shown in Figure 5.8.

Our denoising framework can be easily integrated to the Monte Carlo rendering pipeline as a post-processing stage. The indirect and direct components are outputs of rendering processes. After denoising (see Table 5.3 for an overview of the denoising algorithm), it is sent to other stages for further processing, like tone mapping, data compression, etc. With our denoising technique, a Monte Carlo renderer can use low sampling rates for higher quality image in shorter time.

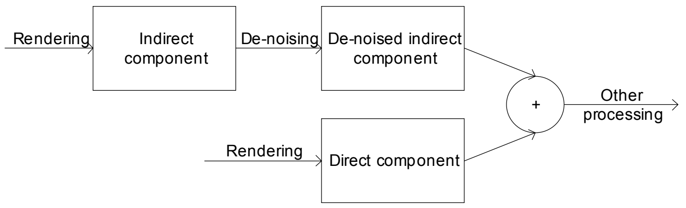


Figure 5.8: Our Denoising Framework Using Bilateral Filtering

5.2.4 Experimental Results

We have implemented our denoising algorithm in C. The direct and indirect components are obtained by adding several lines of code to “rpict” in Radiance (see http://radsite.lbl.gov/) to save the direct and indirect components separately.

Table 5.3: Pseudocode of Bilateral Filtering Denoising Algorithm

```
**Algorithm** MC-denoising
Construct domain filter kernel $c$ with $\sigma_d$;
$\tilde{f} = I * c$; /*convolution for initial estimator*/
For each pixel $p$
        Construct range filter kernel $s$ with $\sigma_r$ and $\tilde{f}$;
        $\kappa = c \cdot s$; /*combine domain and range filters*/
        $\kappa = \kappa / |\kappa|$; /*normalization*/
        $\hat{f} = I(p) * \kappa$;
        Set the value at pixel $p$ with estimator $\hat{f}$;
```

Monte Carlo noise has several ways to contaminate the pixel color, e.g., hue and luminance. Following the approach of [Rushmeier 1994] and [McCool 1999], we assume that luminance channel is most likely contaminated. We compute the luminance for each pixel using Equation (5.18) [Ward 1996].

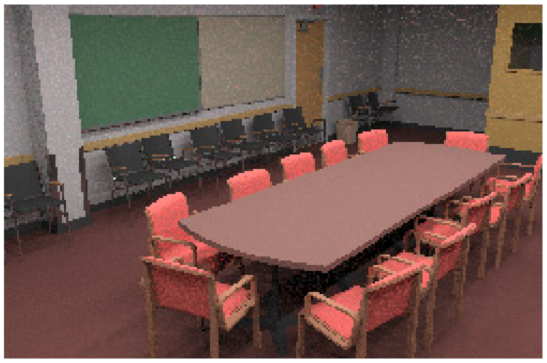
$$I(R,G,B) = 0.265 * R + 0.670 * G + 0.065 * B \tag{5.18}$$

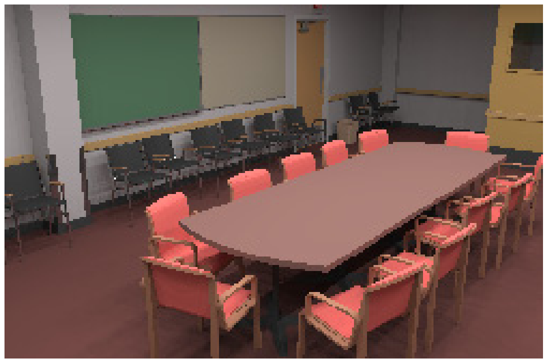
Our denoising results also show that luminance carries most Monte Carlo noise. It is worth mentioning that we carry out Monte Carlo noise reduction in the logarithm domain of the luminance channel. This is because the human eye has a linear response to the logarithm of pixel luminance value.

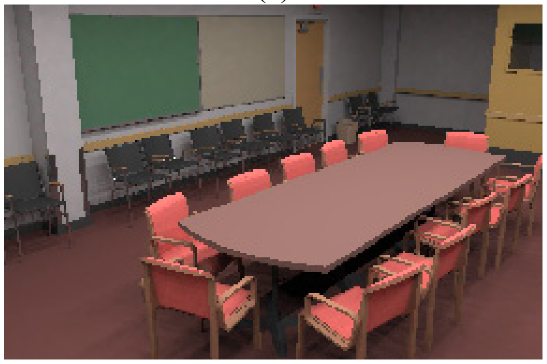
Figure 5.9 and 5.10 show two denoising examples using our Monte Carlo noise reduction operator. More explanation can be found in the caption of Figure 5.10.

Table 5.4 lists time to generate the images in Figures 5.7, 5.9 and 5.10. Our experimental platform is a Celeron 2.0GHz (392M memory, Windows2000). The numbers in parentheses are the number of samples per pixel (sampling rate). In each cell of column 2 and 3, the numbers on the second line are the mean square error (MSE). The denoising time is only a small fraction of the noisy image rendering time, and the time complexity of our denoising algorithm is *O(n)* in most cases, where n is pixel number of the noisy image. We can see that a Monte Carlo renderer using our noise reduction method produces higher quality images in shorter time as compared to producing an image of the same quality by merely improving the sampling rate.

The C source code is available as http://graphics.cs.ucf.edu/mcnr/mcnrBiFilter.c and the executable is available as http://graphics.cs.ucf.edu/mcnr/mcnrBiFilter.exe.


(a)


(c)


(b)

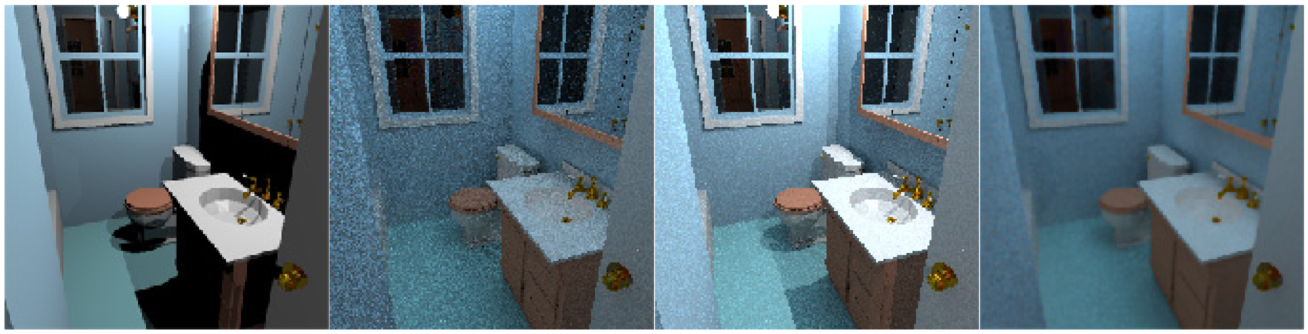
(b) is the denoised result of the image in (a) generated using 5 samples per pixel. This image is very similar to the accurate result shown in (c) which is obtained using 400 samples per pixel.

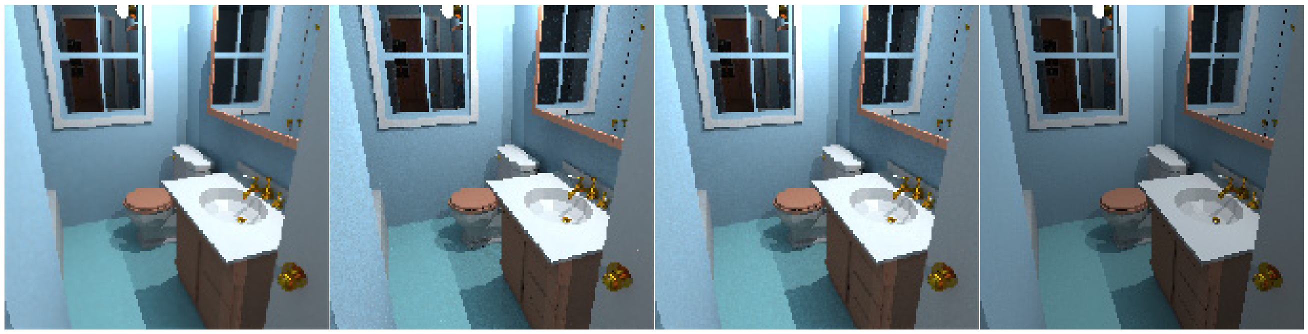
Figure 5.9: Bayesian Denoising of "conference room" Image

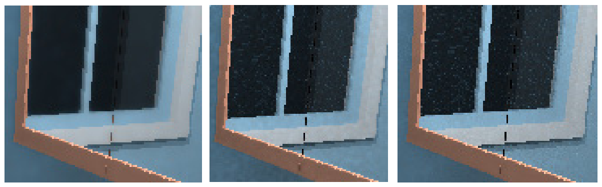
Table 5.4: Statistics Of Bayesian Denoising

	noisy	denoised	accurate	σ_r, σ_d
Living room 400×300	50s(2) 0.152	5.0s 0.089	2100s (500)	2, 0.4
Cabin 512×512	286s(20) .3630	9.8s .2202	3602s (300)	2, 0.4
Conf. room 512×347	183s(5) .0312	6.5s .0275	1802s (400)	2, 0.4

Note: numbers in parentheses denote the sampling rate.


(a) direct component (b) indirect component (c) noisy, 20 samples (d) denoised indirect


(e) our method (f) standard bilateral filtering (g) Wiener filtering (h) accurate, 300 samples


(i) clips from (e), (f) and (g)

(a)-(i) shows the whole denoising process for "cabin" image. (a) and (b) are the direct and indirect components of (c). (b) is denoised using our method to obtain (d). And (e) is the denoising result by adding up (a) and (d). For comparison, we also show the denoising results (f) using standard bilateral filtering, (g) using Wiener filtering. (h) is the accurate image obtained using 300 samples (setting ad=300 in "rpict"). (i) shows clips of the right window on images (e),(f),(g), from left to right. It is apparent that the outliers are removed in (e), but remain in (f), (g). The models of the scene used to generate the images are courtesy of Ward (see http://radsite.lbl.gov/)

Figure 5.10: Some Results of Bayesian Denoising on Image "cabin"

MSE is used as a simple fidelity metric to confirm the relative image quality improvement between the coarsely rendered image and the denoised image. The comparison basis is the image for the same view rendered at very high quality. The MSE measurement is performed using the logarithm of the luminance value, as shown in Equation 7. The larger is the

MSE, the noisier is the image. For the “conference room” in Figure 5.9, MSE of (a) and (b) with respect to (c) is 0.0312 and 0.0275, respectively. And in Figure 5.10, MSE of (c) and (e) with respect to (h) is 0.3630 and 0.2202, respectively. Our denoising algorithm does improve the image quality by reducing the MSE.

5.2.4.1 Parameters Setting

There are two parameters involved in our algorithm: σ_r, σ_d for range and domain filters. In spite of the efforts by Jones et al. [2003], automatic estimation of the bilateral filter parameters remains an open problem. Fortunately, we find $\sigma_r = 2, \sigma_d = 0.4$ are appropriate for most cases of Monte Carlo noise reduction, and we used $\sigma_r = 2, \sigma_d = 0.4$ in all of our experiments.

Although these parameters are only established through experiments, we believe they are closely related to some aspects of human perception including spatial vision and color discrimination.

5.2.5 Analysis of Monte Carlo Noise Reduction using Bilateral Filtering

This section presents a non-iterative local adaptive filter based on bilateral filter for Monte Carlo noise reduction. Unlike other Monte Carlo reduction methods, our approach is able to suppress outliers and inter-pixel incoherence in a unified framework. It can also be used in other denoising tasks, like mesh denoising, where outliers and inter-pixel incoherence coexist. A standard bilateral filtering may be enough in cases where only inter-pixel incoherence needs to be reduced.

The strength of our method lies in its simplicity, robustness and efficiency. It reduces both types of noise in only two passes. The method can be easily adapted to parallel implementation, as well as GPU implementation. We implemented the latter on an ATI RADEON 9700 graphics card which executes the denoising of 512×512 image in a fraction of second. For the "cabin" image in Figure 5.11, our GPU implementation runs at about 2 fps.

This method requires only two parameters σ_r, σ_d. Although further tuning is possible $\sigma_r = 2, \sigma_d = 0.4$ can be used in most cases of Monte Carlo noise reduction. Automatic setting of these parameters is one future research topic.

"Robustness" of our approach lies in the fact that it can effectively suppress Monte Carlo noise in presence of outliers. It can very well suppress inter-pixel incoherence and outliers together.

There are two points about Monte Carlo noise worth to mention. First, this research believes the model parameters to Monte Carlo noise are decided during rendering process, and these parameters can thus be estimated using rendering parameters. Second, the distribution of the Monte Carlo noise varies in different regions of the final synthetic image, and the noise tends to concentrate around where the luminance changes sharply, which necessitates a local adaptive Monte Carlo noise distribution model.

CHAPTER SIX: DYNAMIC OBJECT RENDERING

This chapter presents a pre-computation based method for real time global illumination of dynamic objects [Xu 2004]. Each frame of animation is rendered using spherical harmonics lighting basis functions. The pre-computed radiance transfer (PRT) associated with each object's surface is unfolded to a rectangular light map. A sequence of light maps is compressed using a high dynamic range video compression technique, and uncompressed for real-time rendering. During rendering, we fetch the light map corresponding to each frame, and compose a light map corresponding to any arbitrary, low-frequency lighting condition. The computed surface light map can be applied to the object using the texture mapping facility of a graphics pipeline.

The primary contribution of this approach lies in its pre-computation based real time global illumination rendering of dynamic objects. Spherical harmonics light maps (SHLM) are used to represent the pre-computation results, and the animation can be viewed from arbitrary viewpoints and in arbitrary low-frequency environment lighting in real time. The consequence is an algorithm that is capable of high quality rendering of animated characters in real-time.

The rest of this chapter first discusses the pre-computation process and the real-time rendering process. We then present experimental results. Our final sections present our conclusion and indicate future directions that this work might take.

6.1 Global Illumination Pre-Computation

Our work is built upon the fact that an animation is made up of a sequence of animation frames. Each animation frame constitutes a particular pose of the animation. The PRT for each animation frame is computed. This process generates a huge volume of PRT data for an animation. Fortunately, the PRT is in a form ready for compression if recorded in the parameter space of the object surface. The general process is outlined in Table 6.1.

We make use of a known locality property in our work: neighboring vertices in space and time tend to have similar PRT data, i.e., spatial and temporal coherence applies. This coherence also exists in image/video, and has been successfully exploited for compression. In a similar manner, the use of coherence-based compression techniques can greatly reduce PRT data.

Unlike previous work, our method computes the PRT for each sample in the 2D parameter space of the object surface. In this way, the PRT can be recorded as a 2D "super image"-each pixel consists of an incident radiance spherical harmonics coefficient vector. Thus, the inherent spatial and temporal coherence can be retained and exploited using image/video compression algorithms applied to PRT data compression.

Table 6.1: Outline of Dynamic Objects Pre-Computation and Rendering

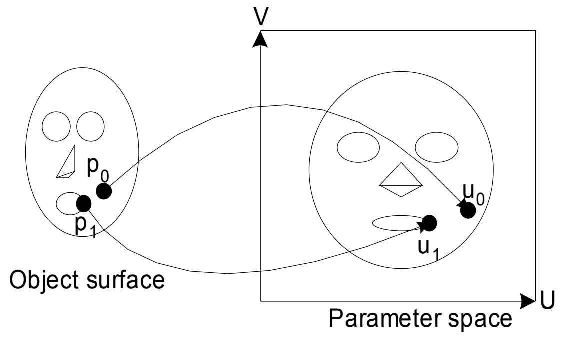
```
Pre-computation phase:
Step A)
 For each animation frame $k$
   For each SH basis lighting $n$
     For each triangle $abc$
        Find $a'b'c'$ in parametric space
        For each pixel $d'$ inside $a'b'c'$
          Find $d$ in object space
          Store PRT of $d$ in $SHLM_n^k(d')$
Step B)
 Compress $SHLM^k$
```
(a)

```
Rendering phase:
Step C)
 During the rendering of frame $k$
   …
   Find lighting L
   Load $SHLM^k$
   Compute SHLM for L
   Feed SHLM to graphics pipeline
   …
```
(b)

(a) is the pre-computation process; (b) is the rendering process.

One key component of our approach is the parameterization of the object surface, i.e., build a one-to-one correspondence between the object surface and 2D parameter space, say [0..1]×[0..1]. For generality and simplicity, we assume that the object surface is made up of triangle meshes. The object surface is unfolded using a mesh parameterization scheme [Gu 2002;

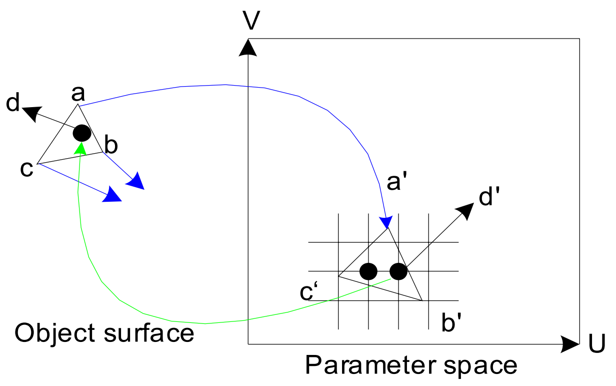
Sander 2002; Alliez 2002; Khodakovsky 2003], so that a one-to-one correspondence is built up between each object surface 3D point and each parameter space 2D point, as shown in Figure 6.1. The 2D parameter space [0..1]×[0..1] is sampled and the PRT of each sample is pre-computed and recorded there.



The left object surface is unfolded to the right parameter space. p_0 is correspondent to u_0; p_1 to u_1.

Figure 6.1: Unfolding Object Surface to 2D Parameter Space

The non-vertex correspondence can be obtained by using barycentric coordinates, as shown in Figure 6.2.

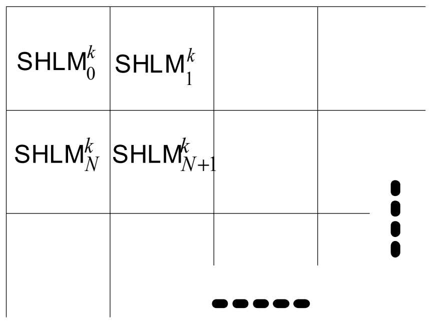


a, b, c are triangle vertices, and *a',b',c'* are mapped triangle vertices in parameter space. Barycentric coordinates of *d'* are used to recover its original surface point *d.*

Figure 6.2: Mapping of Non-Vertex Points Using Bary-centric Coordinates

The incident PRT for each pixel in the parameter space is then calculated by applying the global illumination algorithm with each harmonics basis function as environment lighting [Sloan 2002; Sloan 2003a; Sloan 2003b; Lehtinen 2003]. The result is an "image" with each pixel associated with its incident PRT spherical harmonics coefficients vector. We call this "image" the spherical harmonics light map (SHLM). In the rest of the chapter we use $SHLM_n^k$ to denote the SHLM for animation frame k and spherical harmonics basis lighting Y_n [1].

The sampling points located on the triangle edge can be shared by several neighboring triangles. PRTs for such sampling points are often computed more than once, and the mean of all the results are stored at that point.



The SHLM at row r, column c, will be the $SHLM_{rN+c}^k$.

Figure 6.3: Arrangement of $SHLM_n^k$

6.1.1 Storage Compression of PRT

The amount of raw data from the pre-computation for an animation is huge and to make matters worse the data has a high dynamic range. Fortunately, the image structure of the SHLM, and the spatial and temporal coherence in the data stored in the SHLM lends itself to very high compression. A sequence of SHLMs constitutes a high dynamic range video, and can be

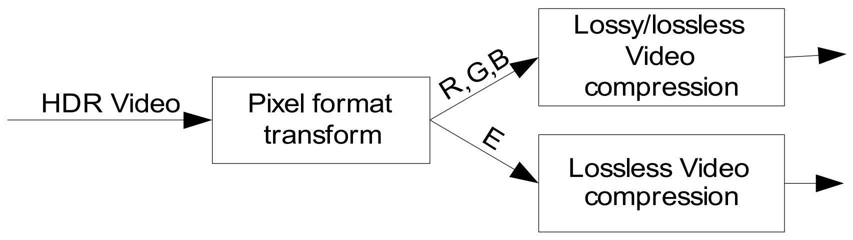
[1] Y_n is actually spherical harmonics basis function Y_l^m where $l = \lfloor \sqrt{n} \rfloor$ and $m = n - l^2 - l$.

compressed by making use of our HDR video compression approach. Details of our HDR video codec are given in the next subsection. It is convenient to tile all $SHLM_n^k$ for the same animation frame together as one SHLMk. If up to N-th order spherical harmonics lighting are used in the PRT computation, $SHLM_n^k$ are placed at location ($\lfloor n/N \rfloor$, $n - N \cdot \lfloor n/N \rfloor$), as shown in Figure 6.3. This results in a total of N×N tiles.

The final SHLMk is sent to the HDR video coder for compression. In our test case, each map is 128×128. The order of the SH used is 2 and this makes a total of 3×3 tiles. Thus the resolution of each $SHLM^k$ is 384×384. Two SHLMs are used to cover the whole body, so the total SHLM size for one animation frame is 384×768. The original data 384×768×4×100 = 118M is compressed to 3.5M in 3.6 min on a Xeon 2.4GHz PC with 1G memory. The decompression cost is 78ms for each frame on the same machine.

6.1.2 HDR Video Compression

The compression scheme is shown in Figure 6.4. The floating point values of the spherical harmonic coefficients for the 3 color channels are stored using the base and exponent used in the RGBE encoding schema [Larson 1991]. This encoding converts the RGB triplet in floating point to a RGBE quadruplet with 8 bits per component. After this encoding, we separate the RGB components to compress them using an existing normal video compression approach, like MPEG [Salomen 2000]. We compress the E component in lossless mode. The decision to use a lossless scheme for the E component is because of the fact that the quality of the decompressed HDR data is very sensitive to the errors introduced in the E component when compressed using a lossy compression scheme.



R,G,B, i.e., color base, and E, i.e., exponential, are separated and sent to different compression schemes.

Figure 6.4: General HDR Video Compression Scheme

The HDR video compression mode used in this chapter, which is implied by Figure 6.4 is:

HDR video lossy compression

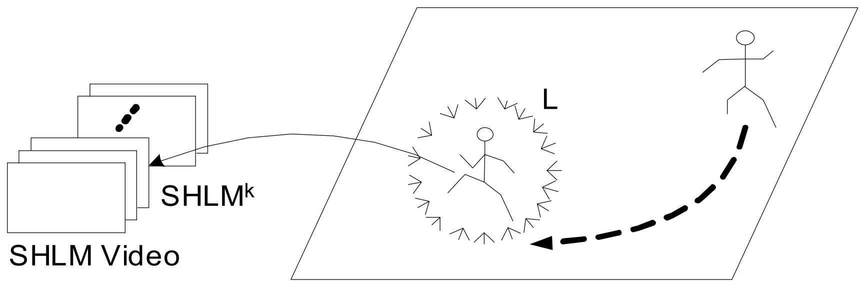
- R,G,B channels: lossy compression
- E channel: lossless compression

The quality of lossy compression for the RGB components is controlled appropriate to available memory budget. An alternative approach worth of investigating uses JPEG2000 [Taubman 2002].

JPEG2000 is a wavelet-based image coding system for various types of still images (like grayscale, multicomponent, etc.). Each of its components supports dynamic range up to 16 bits. It provides a natural way to encode HDR image in lossy/lossless mode through linear quantization. The contrast sensitivity function (CSF) of the human visual system used in JPEG2000 is easy to apply to HDR image encoding. The encoding/decoding of HDR images using the JPEG2000 technique [JasPer 2004] is observed about 2 times slower than that using JPEG technique [FFMPEG 2004], which uses discrete cosine transform (DCT), although the former has much better compression quality in very high compression. We have exploited an efficient HDR still image compression technique using JPEG2000, as described in chapter 4. We are exploring the possibility of developing an efficient GPU implementation of the wavelet decoding scheme for the application of JPEG2000 technique to HDR video.

6.2 Rendering of Dynamic Objects

For rendering the k-th animation frame during the animation, we first compute the SH coefficients L_i, $i = 0,1$ ..., corresponding to the environmental light at the place [Sloan 2002] where our dynamic object is placed.



Current pose is used as index to fetch SHLM^k. Current position of character is used to estimate the environment lighting.

Figure 6.5: Rendering of a Moving Character

The $SHLM_k$ is retrieved from the compressed HDR video. (See section 3.2 for HDR video codec details). The current lighting map is constructed by simply summing up the product of lighting coefficients and $SHLM_k$, as shown below.

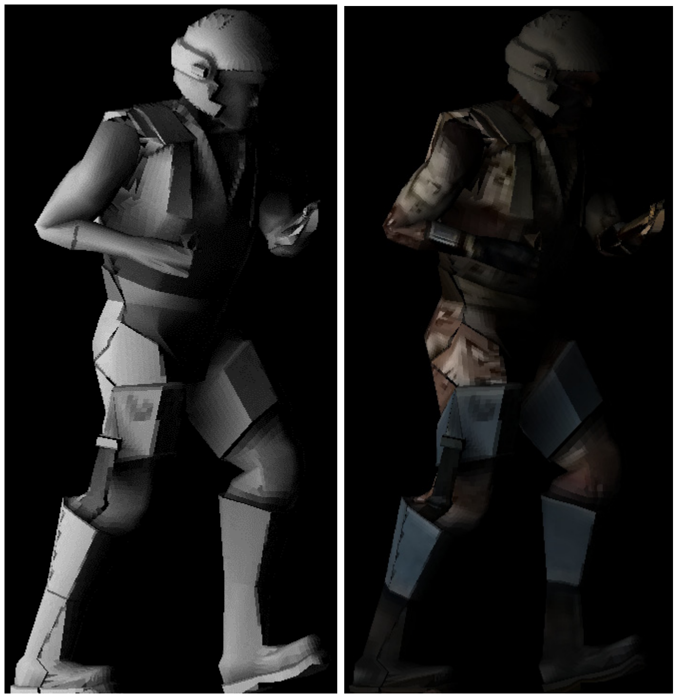
$$SHLM = \sum_i SHLM_i^k \cdot L_i \tag{6.1}$$

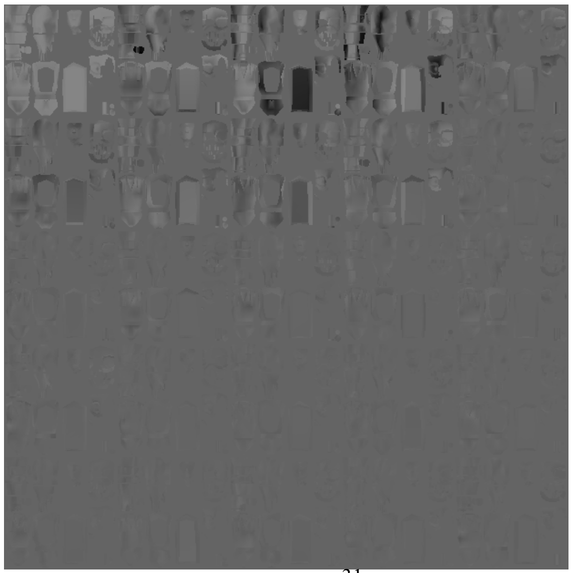
The obtained SHLM can be sent to the graphics pipeline as a texture. The dynamic object is then rendered by texture mapping. See Table 6.1(b) for the outline of the program. Figure 6.5 gives the scenario of rendering.

6.3 Experimental Results

Our experimental subject is a future soldier, an Offensive Force Warrior (OFW), modeled using 3DS MAX 5.0 (The model is courtesy of Media Convergence Lab of University of Central Florida). For a walk action lasting 2 seconds, we extract 100 frames by sampling its pose every 0.02 seconds. Each animation frame consists of 2811 vertices and 2197 triangles. A single frame is shown in Figure 6.6, displayed with lighting (Figure 6.6(a)) and lighting and texture (Figure 6.6(b)). The associated SHLM is shown in (Figure 6.6(c)).

We use the *Radiance* software [Ward 1998] to pre-compute the PRT. The spherical harmonics basis lighting is defined as a "glow" type in the "Radiance" scene definition file. For the parameterization, we make use of the *uv* coordinates generated by 3DS MAX 6.0.


(a) (b)


(c) $SHLM^{31}$

(a) is with only lighting; (b) is with lighting and texture;(c) SHLM for animation frame 31 with SH order up to 4.

Figure 6.6: Some SHLM Experimental Results

The compressed HDR video size varies with the compression quality. Using the highest compression quality, compressed data is about 4.2M. Using middle quality, the size dropped to about 3.5M.

The object surface is diffuse with a reflectance of 0.5.

Since illumination is executed as a texture mapping process, our algorithm is ready for implementation on a GPU. SHLM is assembled in the CPU and sent to the GPU for rendering.

Some statistical data are recorded in Table 6.2.

Table 6.2: Some Statistics of SHLM Experiment

Object	OFW (2811 vertices; 2197 triangles)
Action	2 sec. of walk (100 frames)
Ma6. order of SH	2 (9 coefficients)
Sampling Rate	128 by 128
Pre-computation time of GI computation	1.5 hours for 100 frames
Raw video data	>100 MB (RGBE format)
Compressed HDR video size	3.5M
Rendering speed	>10 frames/sec

Note: The experiment is performed on an Xeon 2.4G with 1G memory running Windows XP.

There are some ways to improve the performance of rendering. First, we can run the decompression step in a separate process from the main rendering, and prefetch the SHLM for the next animation frame. Second, it is possible to feed each compressed SHLM to graphics hardware and decompress it using the texture codec capability of graphics hardware to reduce the traffic between the graphics hardware and the CPU. Third, we can trade some trivial rendering quality by selecting fewer SH basis lighting functions. As shown in Figure 6.6(c), most PRT information converges to the first 9 SH basis lighting functions. We can also send key frame SHLMs to graphics hardware, and compute all the in-between frames by doing interpolation in graphics hardware.

6.4 Analysis

We present a pre-computation based approach for real-time rendering of dynamic objects. The PRT of object surface points are computed and recorded in 2D parameterized space to form a sequence of SHLMs. To save storage this SHLM sequence is compressed using an HDR video compression technique. Dynamic objects are rendered by adding up the products of SHLMs and their lighting coefficients, and then applying the result to the object surface as a texture.

Our approach can perform GI of dynamic objects in real-time, and the objects can be viewed from arbitrary viewpoints and illuminated by arbitrarily low-frequency environment lighting. It is a new way for rendering dynamic objects, but it is restricted to the rendering of predefined actions. Fortunately, this limit can be overcome by combining this approach with motion synthesis techniques, like motion graphs [Kovar 2002].

Compared to [James 2003], our work is suitable for fixed long actions. It is possible to combine our approach with that of James and Fatahalian [2003] to take advantage of the fine dynamics rendering of their approach and of the capability for long actions of our approach.

Recording each PRT as a 2D rectangular image provides several other benefits. Since we record the PRT in 2D parametric space, the size of a PRT is independent of the number of vertices, but depends on only the object surface area and sampling rate. Thus, level-of-detail management is possible. Since our approach keeps the neighborhood of surface 3D points, the coherence between these neighboring points is exploited for data compression. Greater compression rates are achieved by using lossy compression schemes that throw away some high frequency information invisible to the human eye. Lossy compression of HDR image/video is a feasible way to achieve high compression ratio. Finally, this representation of PRT is easily implemented on current graphics hardware as a simple texture mapping.

Our work supports GI features, like self-shadowing and self-reflection to make objects look more realistic, but it cannot create neighboring shadows.

The pre-computed data can be first compressed by applying PCA/CPCA [Sloan 2003b; Lehtinen 2003]. PCA/CPCA is independent of our approach and can be used to reduce the number of dimensions before HDR video compression.

We can use any mesh parameterization scheme. For example, [Sander 2002] gives a signal specialized parameterization, which is a non-uniform parameterization approach that uses more samples wherever there are more details.

Our approach to store PRT has possible applications to other problems. The data from the surface light field [Chen 2002] is of huge volume, and can be reduced by mapping light field data to the parametric space of its surface for compression. It is also possible to enhance the rendering effects by using BTF [Sloan 2003a].

Another way of improving performance is to pre-compute only key animation frames, and interpolate for in-between frames in later rendering. We can also use non-uniform sampling; where the motion is smooth, we use a lower frame rate; where the motion is abrupt, we use a higher frame rate

In some cases where animation blending or inverse kinematics is applied, $SHLM_s$ can be blended or modified accordingly.

In summary, this chapter presents a novel pre-computation based dynamic objects rendering method, which fully exploits the spatial coherence and temporal coherence between neighboring vertices on object surface to efficiently manage the pre-computed data. Our method has potential application in games and mixed reality, which require high quality rendering of dynamic objects in real time.

CHAPTER SEVEN: REAL-TIME REALISTIC RENDERING OF COMPLEX SCENES

Complex scenes are ubiquitous in the real world and practical applications. The issue of their real time rendering thus is worth special research attention. As discussed in the introduction and background chapters, this issue is a big challenge to the computation power of today's personal computers, and new algorithms are desired to fill the gap between the intensive computation required by complex scene rendering and the computation power available by today's graphics hardware.

This chapter presents a new realistic rendering framework for complex scenes, which is based on a novel empirical 3D space subdivision approach. We first describe our new observation on the light transport; followed by the 3D space subdivision method. Our novel rendering framework and our preliminary implementation of this framework is then discussed in next sections.

7.1 Light Transport Analysis

If the whole scene (including geometries and the associated materials) is divided into a collection of local scenes, the light transport will happen either inside local scenes (local light transport), or between local scenes (global light transport). In other words, the light transport happens as either local light transport or global light transport.

In order to clearly discriminate between local light transport and global light transport, let us consider the light transport as a function of radiance along each ray starting from point p_o and reaching point p_i, as shown below,

$L(p_o, p_i)$: radiance starting from point p_o, and reaching p_i.

Assume we subdivide the scene S into a collection of local regions S_n, $n=1,2,..$. *Local light transport* refers to the light transport between points in the same local region, i.e. $L(p_o, p_i)$, $p_o, p_i \in S_n$,. And, *global light transport* refers to the light transport between points from different local regions, i.e., $L(p_o, p_i)$, $p_o \in S_m$, $p_i \in S_n$, $m \neq n$.

Although the above classification of light transport is quite straightforward, it requires further analysis to have some observation that will lead to new rendering algorithms. The light transport taking place in a local scene is first analyzed, and our new observation then presented in the following section.

7.1.1 Lighting Condition Equivalence

Once the light transport in a scene reaches equilibrium, any local scene may be considered as equivalently illuminated by the light field around the local scene (local light field) alone. In other words, any single local scene has the same rendering results illuminated by the light sources in the whole scene as those illuminated by the local light field in the local scene. As shown in Figure 7.1, the local scene is equivalently rendered using light source L_g and light field L_l, which is the surface radiance field on its bounding box.

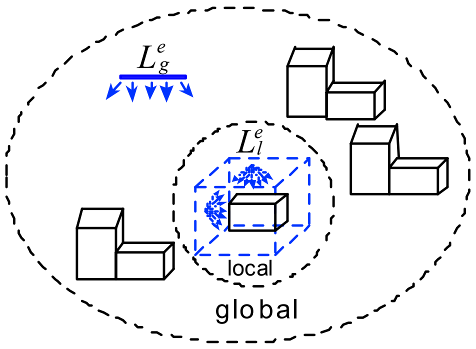


Figure 7.1: Lighting Condition Equivalence.

The lighting equivalence can be mathematically described by Equation (7.1), where the exitant radiance L_o on a local scene along direction w_o is equal to the sum of the self-emission and the integration of all reflected contribution directly and indirectly from every luminary $\vec{x}$ at the light source L_g^e, along path $\vec{p}_g$ in the whole scene with path weight W, and L_o is also equal to the sum of the self-emission and the integration of all reflected contribution from local light field L_l^e at every luminary $\vec{y}$ on the bounding box, along path $\vec{p}_l$ in the local scene [Veach 1997].

$$\begin{aligned} L_o(w_o) &= L_g^e(w_o) + \int_{\vec{x},\vec{p}_g} L_g^e(\vec{x}) \cdot W(\vec{p}_g) d\vec{x} d\vec{p}_g \\ &= L_l^e(w_o) + \int_{\vec{y},\vec{p}_l} L_l^e(\vec{y}) \cdot W(\vec{p}_l) d\vec{y} d\vec{p}_l \end{aligned} \qquad (7.1)$$

So, once the local light field is known, the local scene can be rendered by simulating only a local light transport. If the local scene is static, pre-computed radiance transfer (PRT) based techniques can be applied for real-time realistic rendering of the local scenes.

The above analysis leads to a novel rendering framework, in which local light fields are first computed, and local scenes are then rendered using their local light fields as light sources. The breaking up of light transport process into global light transport and local light transport simplifies the computation. For real-time rendering precise computation and representation of local light fields is neither affordable nor necessary. In this study, the radiance on eight vertices of the bounding box is used to approximate the local light field, and a tri-linear interpolation is used to compute the radiance on other points in the local region. The error in the local light field approximation and the rendering time are balanced by our empirical 3D space subdivision approach.

The validity of this approximation method is justified through many observations. First the local light field off the scene surface generally changes slowly. Second, the high-frequency shading effects mostly come from scene BRDF, normal perturbation, local occlusion, and local reflection. This may be the reason why environment lighting is considered a feasible lighting condition approximation, and thus receives much research effort. However, environment lighting assumes far-field light field. In our rendering framework, local scene may not be far from its neighboring scene and light sources, thus a higher order function should be used to approximate the local light field. Higher order means higher cost. We use spherical harmonics to represent the incident radiance function around a point, and represent the local light field using linear interpolation. Nice spherical harmonics basis functions are used to approximate the incident radiance function on each point in our preliminary implementation.

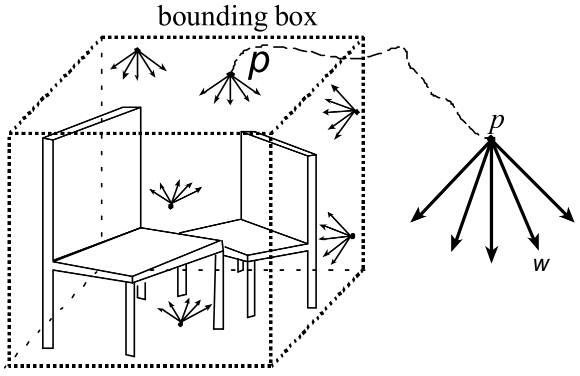


Figure 7.2: Local Lighting Condition

7.1.2 Local Light Field Approximation

We define the local lighting condition as the incident radiance field over its bounding box, as shown in Figure 7.2. It is a 4 dimensional function (2 dimensions to describe the point position on the bounding box, and 2 dimensions to describe direction), as shown in,

$$L_{LLF}(p,w) \text{ where } p \in BB, w \in \Omega,$$

where, p is point on the bounding box BB and w is the direction over sphere Ω towards point p.

It is convenient to approximate the local lighting condition using the incident radiance field on 8 vertices of the bounding box, which are approximated using spherical harmonics, as shown in Figure 7.3. As long as the light fields on the 8 vertices are available as L_A, L_B, L_C, L_D, L_E, L_F, L_G, L_H, the light field at any point can be tri-linearly interpolated. If we denote the bary-centric coordinates as (x_A, x_B, x_C, x_D, x_E, x_F, x_G, x_H), then the light filed at any point (u,v,w) is tri-linearly interpolated as Equation (7.2). The bary-centric coordinate of some point for each vertex is computed as the volume of the rectangular solid determined by its diagonal connecting this point and the opposite vertex of this vertex.

$$\tilde{L}(u,v,w) = \sum_{i=A,B,C,D,E,F,G,H} x_i L_i \tag{7.2}$$

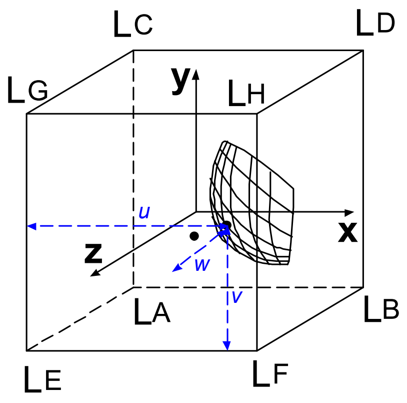


Figure 7.3: Cubic Barycentric Coordinates for Tri-linear Interpolation

This research defines a metric to measure the local lighting condition approximation error, as shown in Equation (7.3).

$$\varepsilon \equiv \int_{x \in BB, w \in \Omega} \left(L(x,w) - \tilde{L}(x,w) \right)^2 dx dw \tag{7.3}$$

By subdividing the bounding region or merging itself with the neighboring bounding region, ε can always be reduced to within some tolerance. Although it is possible to find a subdivision of the scene by this way, it is not a practical method in real applications. One reason is that the light field is unknown in advance; another reason is that ε is expensive to compute. We instead propose a practical 3D space sampling approach to find a collection of bounding regions (samples) that can enclose the whole scene. These samples will be able to linearly approximate the light field with visually acceptable error.

7.2 Practical 3D Space Subdivision

The subdivision approach finds a balance between two factors: rendering time and rendering accuracy. The performance depends on the geometric complexity of local scene. The geometric complexity is closely related to the computation efforts in rendering. The more complex is the local scene, the more time is required in local light transport computation. But if there are too many local regions, too much time will be required in computing local light field approximation. The rendering accuracy depends on that of the local lighting condition approximation. By controlling the minimum mean distance to its neighboring scene, a neighboring complexity metric can be used to control the local light field approximation error.

7.2.1 Geometric Complexity

It is a still an open problem to accurately estimate the rendering time. We propose a metric to approximate the rendering time based on geometric information. We build our geometric complexity metric X_i^G as a multiplication of local surface area and local folding degree, as shown in Equation (7.4). The folding degree is the ratio between scene surface area and bounding box surface area. More geometry inside a region, higher is the folding degree.

$$\mathrm{X}_i^G \equiv \frac{S_i}{\sum_k S_k} \cdot \frac{S_i / D_i^2}{\sum_k S_k / D_k^2} \tag{7.4}$$

where S_i is the surface area of scene geometries in local region i, and D_i is bounding box size of local region i. The denominations are normalizations.

By constraining each local region to the same geometric complexity, the computing effort is uniformly distributed to each local region.

7.2.2 Neighboring Complexity

This research defines the neighboring complexity X_i^N as the reciprocal of area weighted geometric mean distance to neighboring scene, as shown in Equation (7.5).

$$X_i^N \equiv \frac{1}{\sum_j A_j} \cdot \sum_j A_j \frac{1}{l_{i,j} / R_i} \tag{7.5}$$

where, $l_{i,j}$ is the distance from primitive j to the center of the neighboring scene with bounding box size (diagonal length) R_i , and A_j is the area of primitive j. The geometric mean of distance to neighbors is used in order to favor smaller distances.

This metric is used to determine if the neighboring scene to some local scene is far enough so that the local lighting condition can be linearly approximated.

With these two complexity metrics, it is possible to sample the 3D space from the bounding box of the whole scene, and subdivide it until both geometric complexity and neighboring complexity are below some user defined thresholds.

7.2.3 Experimental Results

This research uses the data structure *octree* in the 3D space subdivision implementation.

Figure 7.4 shows some examples. Local regions are drawn in blue. Figure 7.4(a) is the 3D sampling result to the scene Figure 7.4(b) and (c) is for the scene in Figure 7.4(d). Notice where the scene geometry is denser, there is denser subdivision.

There are two used specified thresholds for geometric complexity and neighboring complexity. This research uses 0.05 and 0.3 respectively in the experiments.

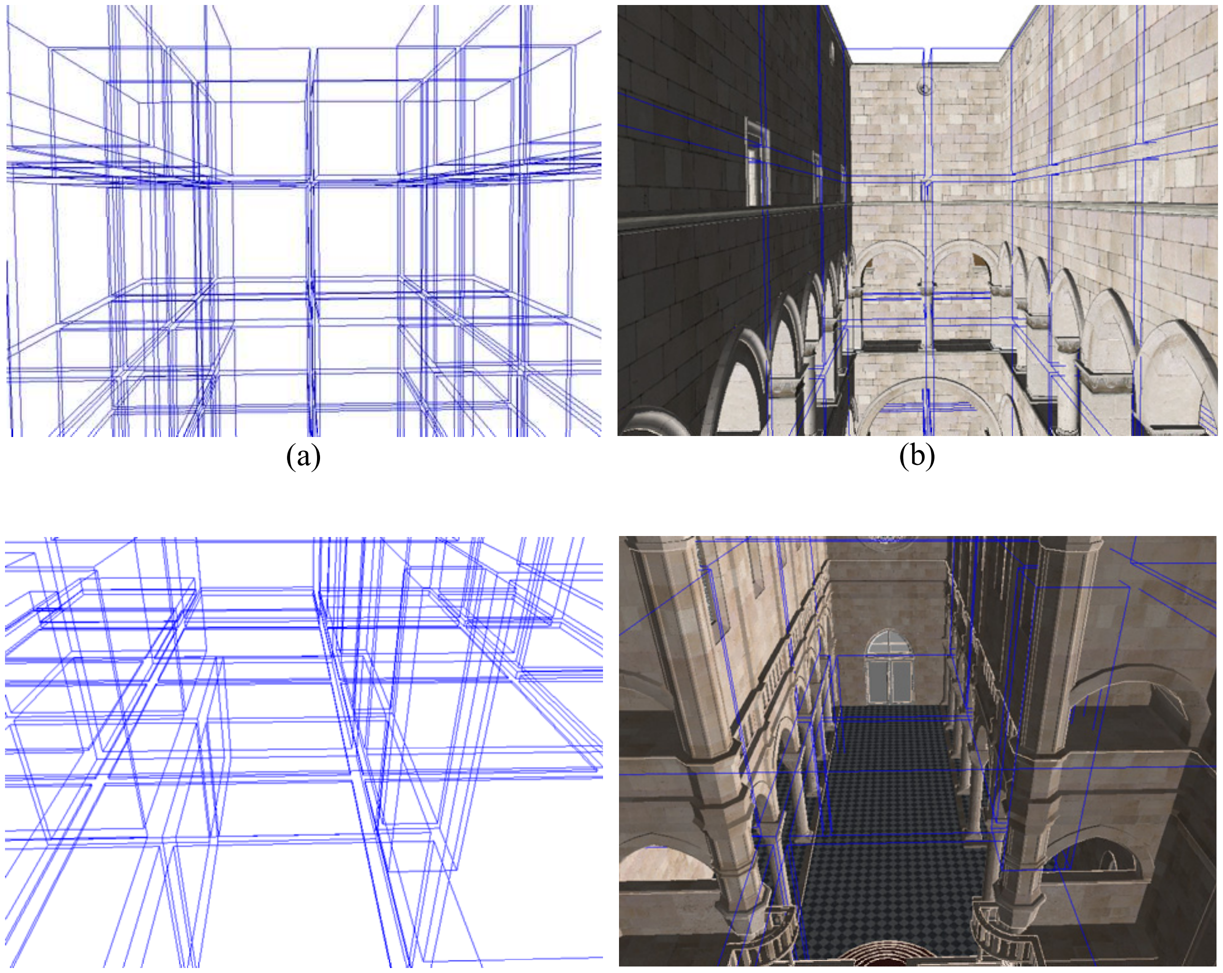


Figure 7.4: 3D Space Subdivision

7.3 Interactive Global Illumination Walkthrough

The local light field approximation leads to a realistic rendering method to support real-time rendering of complex scenes. The whole process as shown in Figure 7.5 is composed of three steps: 3D space sampling; compute local light field approximation; render using local light field approximation. The global rendering process resumes at any lighting change.

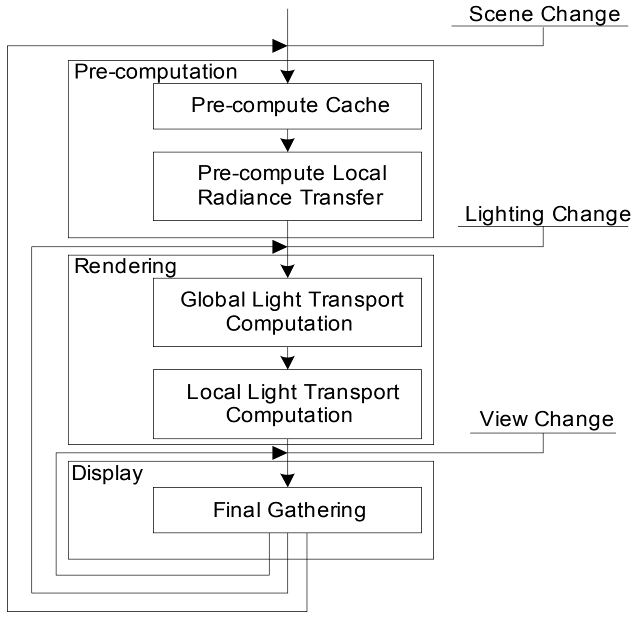

Scene Change
Pre-computation
Pre-compute Cache
Pre-compute Local Radiance Transfer
Lighting Change
Rendering
Global Light Transport Computation
Local Light Transport Computation
View Change
Display
Final Gathering

Figure 7.5: Rendering Algorithm Using 3D Space Subdivision

For real-time local rendering, this research makes use of the PRT under near-field illumination [Heidrich 2000; Kautz 2004] with the local light field interpolated linearly instead of using spherical harmonics gradients.

The global light transport step distributes radiance to local regions by simulating Equation (7.6).

$$\begin{aligned} &L_0 = L_e \\ &L_n = L_e + TL_{n-1} \\ &\lim_{n\to\infty} L_n = L \end{aligned} \tag{7.6}$$

where, L_e is the initial lighting condition, L_n is the lighting distribution after n bounces, T is the light transport operator, and L is the light distribution at equilibrium state.

In the first iteration of the global light transport, each local region receives lighting contribution from the light source, and encodes it as spherical harmonics coefficients. And in the following iterations, each local region then receives lighting contribution from other local regions. See Table 7.1(b). This method simulates only the direct contribution and first bounces in the implementation, because multiple bounces contribute visually little significant rendering features but require extensive computation.

Table 7.1: Pseudocode of Rendering with 3D Space Subdivision

```
(a) Compute $L_o$
1 get direct contribution $L_{i,direct}$
2 interpolate for $PRT$
3 interpolate for $L_{i,indirect}$
4 return $Lo=(L_{i,direct}+L_{i,indirect}) *PRT$
```

```
(b) Global rendering
1 for each bounce
2   for each caching region
3     for each vertex
4       for each direction
5         compute hit point
6         compute excitant radiance of hit point $Lo$
7     store incident radiance field in terms of SH
```

```
(c) Local rendering
1 for each view ray
2   find its first intersection point with the scene
3   compute $Lo$ at the hit point using procedure (a)
```

Based on the local light field approximation, we can render the scene for some specific view point by simulating the light transport in local region, as shown in Table 7.1(c).

We implemented point light source and diffuse material in our experiments, though our rendering framework is not limited to any specific geometry of the light source, and any specific material property. Some results are shown in Figure 7.6. “sponza” (Figure 7.6(a)) has 108K triangles, and renders at 10 fps. Pre-computation takes 20 mins and lighting change takes 0.5 sec. “sibenik” (Figure 7.6(b)) has 105K triangles and renders at 12 fps. The Pre-computation takes 25 mins and lighting change takes 1 sec. The tests are run on a Dell desktop computer (1.7G Xeon CPU, 1G memory, Windows XP). The test scenes are courtesy of Marko Dabrovic from www.RNA.HR.

7.4 Analysis

Our method is an extension of the PRT approach by supporting lighting inside the scene. It is a hierarchical rendering approach, where the global pass simulates the light transport in a global scale, and the local pass simulates the light transport in local scenes.

The approach provides near real-time performance for computing plausible results rather than physical accuracy in either the local light field approximation or the light transport computation. The accuracy depends on many factors: order of spherical harmonics in approximating the incident light field; order of function in approximating the local light field; bounces in global light transport simulation; and error from local PRT. A simple increase of spherical harmonics orders or light transport bounces increases the accuracy at the expense of interactive performance.

This research provides the possibility to build a fast hierarchical algorithm for realistically rendering complex scenes, by recursively applying the 3D space subdivision algorithm to large local scenes. It is also possible to implement the global light transport and local light transport of our rendering framework in programmable graphics hardware for a higher performance. The rendering error of our approach needs further study to apply it to physically accurate rendering, which raises more issues: 1. error analysis of local light field representation; 2. accurate distribution of light to local light fields; 3. accurate local light transport.

(a) "*sponza*"

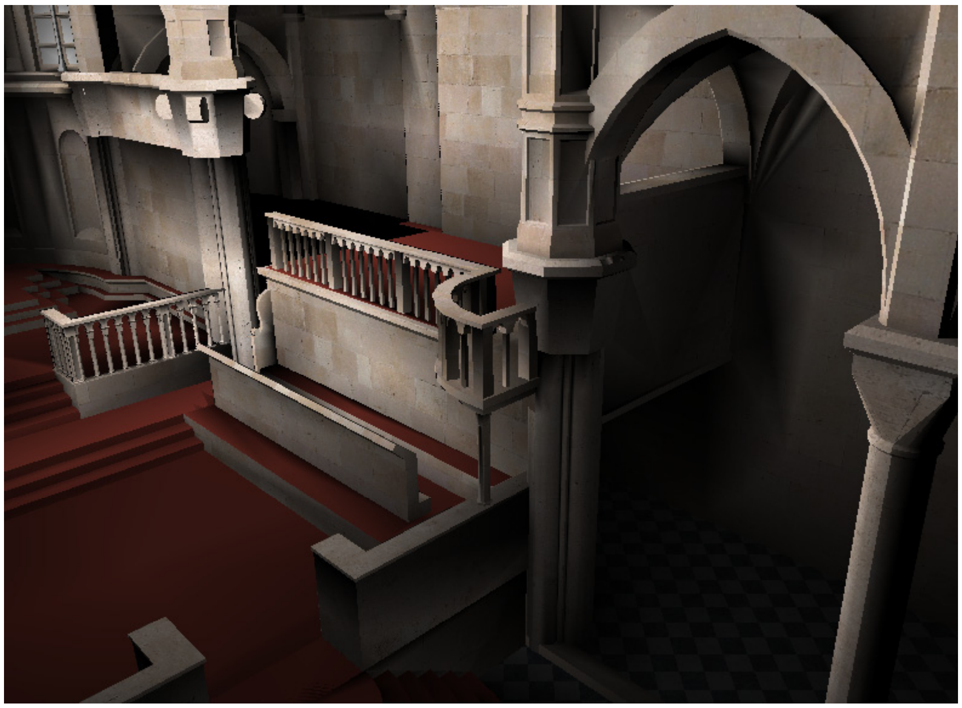

(b) "*sibenik*"

Figure 7.6: Rendering Results of Complex Scenes

Although it is convenient to approximate local light fields through spatial subdivision using *octtree*, a non-regular subdivision can be optimal in the sense that the scene light field can be accurately represented using the fewest samples as possible. A possible idea is to first distribute the scene surfaces into clusters so that the ratio of the mean distance to neighboring clusters of some cluster and the size of this cluster is no less than some threshold, and then find the 3D Voronoi tessellation of clusters for light field subdivision.

In summary, this chapter proposes a practical 3D space sampling algorithm and applies it to a realistic walkthrough framework. It supports interactive lighting change, and real-time walkthrough rendering. The preliminary implementation has demonstrated this novel rendering framework can support real-time realistic walkthrough of complex scenes on a desktop computer.

CHAPTER EIGHT: APPLICATIONS OF RENDERING IN MIXED REALITY

Mixed Reality (MR) [Milgram 1994] research deals with problems related to seamless integration of real and virtual for providing immersive experience in many practical applications. Of various issues related to this problem, our work deals with issues dealing with seamless visual integration. We believe that since visual information dominates the human perception accurate visual integration should be of primary concern.

Before describing the visual integration issues and our solutions, we would like to briefly discuss the MR platform on which our algorithms run. The MR platform [Uchiyama 2002] used in this research is schematically illustrated in Figure 8.1. The video see through head mounted display (HMD) allows us to see the real world captured through a pair of tiny video cameras placed in front of our eyes. The captured video is fed to a pair of tiny LCD displays placed in front of the eyes, in between the eye and camera. The device provides us with the capability of mixing virtual rendering from the computer with the live video data before it is fed to the display. The sensor system, composed of transmitters and receivers, allows us to track the position and orientation of the HMD and hence track the position and view direction of the human observer wearing the HMD. The sensor system provides us with a physical means to geometrically align the virtual and the real world. The connections between all components are shown in Figure 8.1.

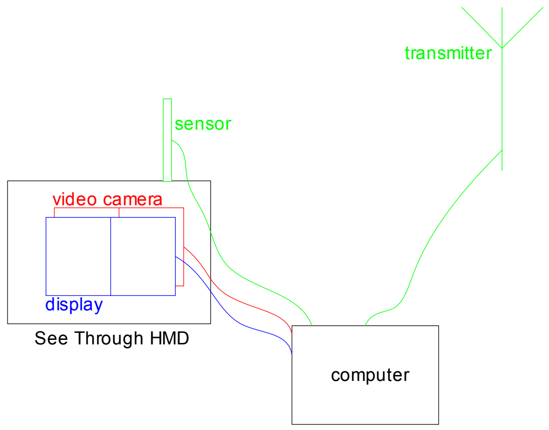


Figure 8.1: MR System Research Platform

8.1 Visual Integration Issues in Mixed Reality

There are three main issues related to accurate visual integration of the real and virtual in a mixed reality world. They are geometrical alignment, visibility, and light transport. Geometrical alignment determines the relative position of the virtual objects in the real world, or that of the real objects in the virtual world. Visibility determines the relative position of virtual objects and real objects, and thus allows us to determine what parts of a virtual object are occluded by real objects and vice versa. Light transport deals with the direct lighting, shadows, and inter-reflection of light between virtual objects and real scene.

In our system geometric alignment is addressed physically using a sensor system. Although vision based algorithms for computing visibility of the real world objects with respect to the camera are available, at the time of this work no real time hardware or software depth recovery method was available to the MR community. So, like many other MR applications, we assumed that some form of geometric model of the real world was available to the MR system. In the absence of such models conventional practice is to superimpose virtual objects on the real scene. The work presented in this chapter addresses issues related to the last component of improving visual integration, that of light transport between virtual world and real world. The issues related to light transport can be categorized into two classes: *illumination* and *shadow*. By *illumination* we mean:

- lighting of virtual objects by real world illumination,
- lighting of real objects by virtual light source(s), and
- inter-reflection between virtual world and real world.

And by *shadow* we mean:

- shadows cast from virtual objects to real world,
- shadows cast from real objects to virtual world.

We have addressed two of these sub-issues: lighting of virtual objects by real world illumination and shadows cast from virtual objects to the real world. The remainder of this chapter discusses our methods to attack these two issues.

8.2 Virtual Object Rendering and Shadowing

Many mixed reality applications insert virtual objects into the real world. To make the virtual object look an integral part of the world, we should render the virtual object as if illuminated by the same lighting that illuminates the real world. We also need to add any shadows generated due to the insertion of virtual objects between real lighting and the background. We present a solution to solve the former issue by integrating some known algorithms, and propose a novel means of incorporating dynamic virtual objects into the real world. We also introduce a pre-computation based method to generate and add the soft shadows of virtual objects to the real background.

8.2.1 Rendering of Virtual Objects Using Real World Lighting

The very first step in illumination using real world light is to capture this light. In the real world light comes from everywhere in the scene. We use an environment capture video camera, Lady Bug [http://www.ptgrey.com/] to capture the environment light from a position in the real scene in the neighborhood of where the virtual object will be inserted. The camera captured data is of low dynamic range (LDR). Using the multiple exposure method proposed by Debevec [1997] we convert the LDR environment data to HDR data. Thus, the captured light is a close approximation to the lighting condition around the region of interest. We use this environment light to illuminate the virtual objects.

The captured environment light may be considered as an incident radiance function over the angular space around a point, as shown in Equation (8.1).

$$L : [0,\pi] \times [0,2\pi] \rightarrow [0,\infty] \tag{8.1}$$

Lighting of any point on the virtual object is the integration of the incident radiance function and the surface reflectance property. This integration expression is shown in Equation 8.2.

$$L_o(w_o : \vec{n}) = \int_{\Omega} L(w : \vec{n}) f_r(w_0, w) \cos\theta dw \tag{8.2}$$

where $L(w : \vec{n})$ is the incident radiance of environment lighting from direction w relative to the surface normal and $f_r(w_0, w)$ is the surface BRDF. Computation of an integral over the hemisphere is normally carried our by using a Monte Carlo quadrature technique. Such techniques are computationally expensive and hence are not suitable for the real-time requirements of MR applications. We avoid Monte Carlo quadrature by using a recently developed function approximation based technique.

8.2.1.1 Rendering of Static Virtual Objects

We adopt the *environment map rendering* technique [Ramamoorthi 2001, Ramamoorthi 2002] to render the virtual objects using the captured environment lighting. This technique can render the virtual objects in real time by transforming the integration equation (Equation 8.2) into a vector dot product of lighting coefficients and BRDF coefficients. We approximate the captured radiance function into a coefficient vector using spherical harmonics basis set. The equation for computing the coefficients is as follows.

$$l_n = \int_{\Omega} L(w) \cdot b_n(w) dw$$

where, $L(w)$ is the incident radiance of environment lighting from direction w with respect to a global axis; b_n is the $n'th$ spherical harmonics basis function; and l_n is the environment lighting coefficient corresponding to spherical harmonics basis function b_n.

The BRDF is similarly approximated by a spherical harmonics basis function to get its coefficients $\vec{f}_r$. Due to the orthonormality property of spherical harmonics basis functions, the environment lighting rendering equation is transformed to a vector product, as shown in Equation (8.3).

$$L_o(w_o : n) = \int_{\Omega} L(w : \vec{n}) \cdot f_r(w) \cdot \cos\theta_i dw = \left(M(\vec{n}) \cdot \vec{l}\right) \cdot \vec{f}_r = \vec{l} \cdot \left(M(-\vec{n}) \cdot \vec{f}_r\right) \qquad (8.3)$$

where, $L_o(w_o : \vec{n})$ is the radiance toward viewer in direction w_o at some vertex with normal $\vec{n}$; $M(\vec{n})$ is the spherical harmonics rotational matrix that converts the environment lighting coefficients from world coordinate system to local coordinate system, and $M(-\vec{n})$ vice versa.

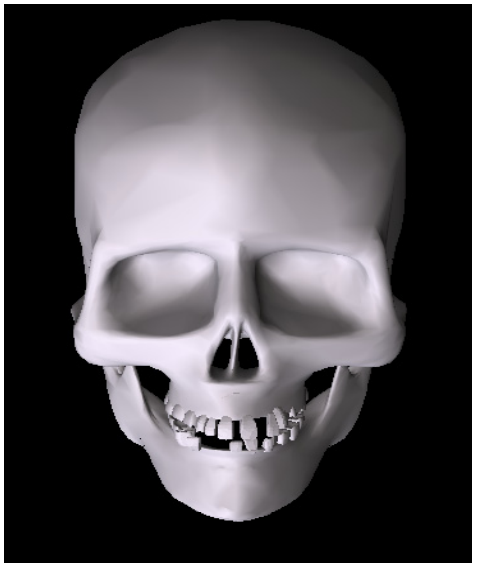


Figure 8.2: An Irradiance Rendering Example

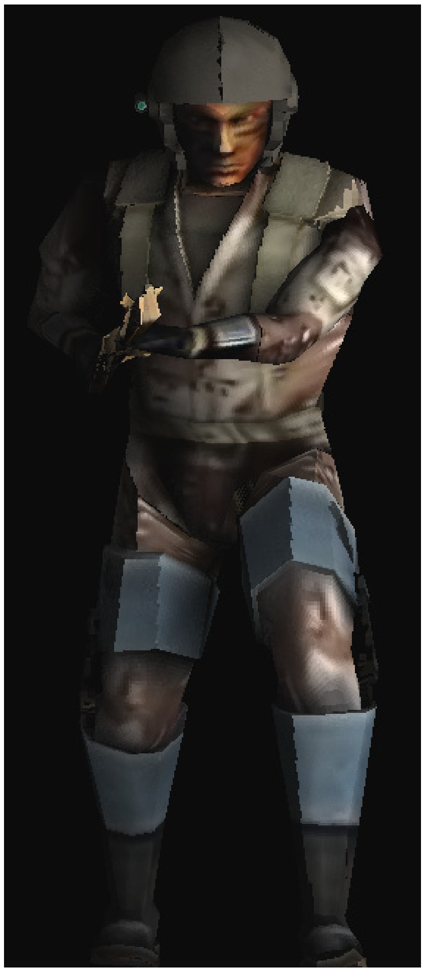
Figure 8.2 is a virtual object rendered using the environment lighting inside UCF's Media Convergence Laboratory. We use 9 spherical harmonics basis functions in our experiments. Because the rendering is reduced to a simple vector dot product problem, Equation (8.3) can be implemented using programmable graphics hardware for further performance improvement. The PRT techniques [Sloan 2002] can enhance the visual realism of virtual objects rendering by accounting for self occlusion and self inter-reflection of the static virtual objects. We extend the PRT for the real time rendering of dynamic virtual objects.

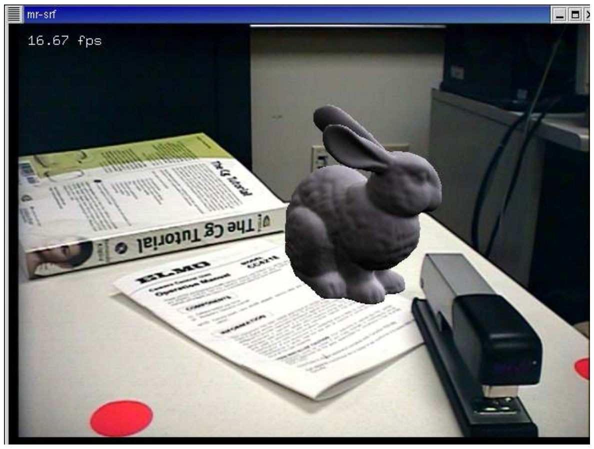
8.2.1.2 Rendering of Dynamic Virtual Objects

Use of dynamic synthetics objects is very common in MR applications. Using the algorithms described in Chapter 6 we create realistic renderings of virtual dynamic objects in real time. This technique pre-computes the radiance transfer for each key frame of the animation, and applies a linear combination of the pre-computed data in the rendering stage. Figure 8.3 is a snapshot of a walking “offensive force warrior (OFW)” rendered using pre-computed radiance transfer data. The soft shadow on the right leg cast by the left leg of the soldier improves the visual realism of the virtual dynamic objects.

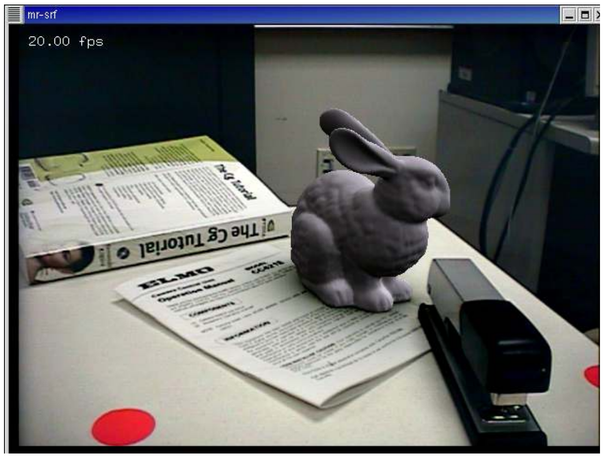
8.2.2 Shadow from Virtual Objects to Real-World Background in Real Time

Shadows anchor an object to the space. Computing shadows is thus essential for the visual integration of synthetics objects in the real world. As we see in Figure 8.4(a), without shadows virtual objects inserted into the real world stand out as if floating in space. In this section we present our soft shadow generation technique for mixed reality applications. We first pre-compute the shadow of the virtual object on a phantom receiver for each spherical harmonic basis environment lighting. During the actual rendering time we compose the shadow for the captured light of the real environment by summing up the pre-computed shadows, and then mapping the composed shadow from the phantom background to the real background using the differential rendering method introduced by Debevec [1998].





(a) without shadow



(b) with shadow

Figure 8.3: Real-Time Rendering of "OFW"

Figure 8.4: Soft Shadow from Virtual Object to Real Background

The phantom shadow receiver is assumed to be a planar surface immediately under the virtual objects. We represent the basis shadow as an HDR image, which is calculated as Equation (8.4).

$$PSM_n = IMG_{noobj,n} - IMG_{obj,n} \tag{8.4}$$

where, PSM_n is planar basis shadow corresponding to the n^{th} spherical harmonic basis function as the lighting; $IMG_{noobj,n}$ and $IMG_{obj,n}$ are the intensity maps on the receiver due to n^{th} spherical harmonic basis lighting without and with the object of interest.

As done in earlier sections, the captured environment light due to the real world is first converted into spherical harmonic coefficients $\vec{l} = l_0, l_1, \ldots$, and the shadow *PSM* under environment lighting is computed by a weighted summation of the basis shadows, as shown in Equation (8.5).

$$PSM = \sum_n l_n \cdot PSM_n \tag{8.5}$$

The *PSM* is finally mapped into the real background just before superimposing the virtual objects.

The result shown in Figure 8.4(b) is due to 9 spherical harmonics basis functions. The “bunny” seems to be in direct contact with the table because of the shadows. We implemented the shadow composition using the pixel shader of the GPU for real time performance.

8.3 Discussion

In this chapter we presented our work related to realistic lighting for mixed reality. The work includes accurate rendering of virtual static and dynamic objects in real time using live captured real lighting, and computation and rendering of soft shadow of virtual objects on the real background in real time. We conclude this chapter by discussing the limitations of our methods. Though the HDR video camera is a convenient mechanism to capture the real environment lighting at a discrete number of places, it is intrusive and the captured data may not be representative of the actual illumination on every surface point of large virtual objects. Our shadow generation technique for virtual object is limited to computing the shadow for the upright object on a plane below the surface. Shadow due to change of orientation of the objects is not captured correctly. Further, mapping the shadow from the planar phantom to irregular receiver surfaces is not trivial. The key to advancing the light transport issues in mixed reality is the automated scene recovery of the dynamic 3D world. This is essential to accurate placement of synthetics objects into the real world with occlusion and shadows.

CHAPTER NINE: CONCLUSION AND FUTURE WORK

Light transport and HDR image processing are integral components of the rendering framework shown in the introduction chapter. All the work presented in this thesis addresses problems related to these components. Complex scenes and dynamic objects are two classes of scenes common in real world applications, but rendering them is computationally expensive. This thesis proposes an efficient complex scene rendering approach based on an empirical 3D space subdivision method, and a real time dynamic object rendering technique by pre-computing the radiance transfer of each animation frame. The light transport simulation using Monte Carlo algorithms suffers from noise artifacts. This thesis explores a Bayesian method and a bilateral filtering method to suppress Monte Carlo noise in synthetic images. Most synthetic images are naturally HDR images. The visual display of HDR images on conventional LDR devices, and data compression of HDR images for efficient storage and transmission are two important areas of research. We propose a novel tone mapping algorithm using the level set method, and two new HDR image compression algorithms based on DCT method and JPEG2000 framework.

The research results presented in this thesis contribute to bringing rendering, particularly real time global illumination, to real world applications, making wider application of HDR images possible.

In the rest of this chapter we first broadly discuss major directions of future study of light transport. We then present our proposals that arose from the research described in the previous chapters. The new proposals include cache based visibility function computation; perceptual based HDR image compression; and a novel data format of minimum number of bits.

9.1 Future Directions of Study

The quest for *realism in real time* has always been the driving force behind rendering research. We see adaptive rendering and perception based rendering as the two most important future research directions in improving rendering efficiency.

Adaptive Rendering breaks the rendering task into many subtasks, and automatically chooses appropriate rendering algorithms for different rendering subtasks. Most of the currently available rendering algorithms are only efficient for a few very specific test cases. A clever rendering algorithm should be able to make adaptive choices of the most suitable algorithm depending on the complexity of the task on hand.

Perception Based Rendering exploits the knowledge of the HVS to guide the rendering algorithms for efficient rendering. We believe that further study of visual perception in rendering specific problem will help us simulate realism much more efficiently that what is possible now.

In recent years, advances in technology have greatly reduced the cost of digital imaging. It is now easy and inexpensive to capture images that are rich in real world information: geometry, material, and lighting. This has inspired researchers to use images as input to rendering algorithms (image based rendering or hybrid rendering), to study the light transport between virtual 3D objects and real objects in 2D images (light transport between virtual objects and real objects), and to extract material information from the images (inverse rendering).

Image Based Rendering makes use of the information in images to render a new image. Images have been used to represent 3D models for rendering. They are used to interpolate images to create new views. Single images are used as the sources to modify some elements of target images. Consistent lighting and geometric registration are still major challenges. Moreover, a theoretical basis of image-based rendering has yet to be developed.

Light Transport between Virtual Objects and Real Objects is a fundamental problem in many new mixed reality and augmented reality applications. The integration of virtual and real objects has shown many promising applications. Consistent illumination, shadows between them, is essential for seamless integration of real and virtual.

Inverse Rendering promises to recover much important information from the real world (BRDF, geometry, lighting, and surface appearance) from images. This information can in turn be used in new rendering. These acquired elements bring diversity of natural scenes and realism to rendering.

The following section presents some possible future work that follows these directions of research.

9.2 New Research Proposals

9.2.1 Cache Based Visibility Function Computation

Environmental maps are popularly used as light sources in image based lighting, which is a simple and effective way to introduce real world lighting to rendering. Spherical harmonics makes possible real-time rendering with environmental lighting. The pre-computed radiance transfer (PRT) technique improves rendering realism of environmental lighting by pre-computing the self occlusion and self inter-reflection and representing the pre-computed data in spherical harmonics basis. However, pre-computation is time consuming. Kautz proposed a hemispherical rasterization method to expedite the pre-computation step [2004], but Kautz's method is limited to scenes no larger than several thousand triangles. We propose to extend Kautz's method to larger scenes by caching visibility functions.

Some parts of the surfaces of a concave object are occluded from environment lighting by the object itself, and the self occlusion produces self-shadows. Simulation of self shadows can significantly enhance the visual realism of the rendered scene, but the shadow due to environmental lighting requires much computation. The visibility function between a surface point and the environment lighting decides how much of the environment lighting contributes to the illumination of the surface point. For each surface point, its visibility function to the environment lighting along each direction, $v(\theta,\phi)$, is a binary function of direction over its hemisphere. A table-lookup based visibility computation method is proposed in [Kautz 2004] to enable such real time self shadow computation for objects made up of no more than a few thousand triangles. To overcome the difficulty in rendering objects with more triangles in real time, it is possible to exploit the spatial coherence of visibility function by a caching technique. The neighboring points tend to have similar visibility function. We aim to render tens of thousands of triangles in real-time with self-shadow due to environment lighting. Such work is very useful in mixed reality applications. The virtual objects that are integrated into real scene are rendered using the real lighting with convincing soft shadows.

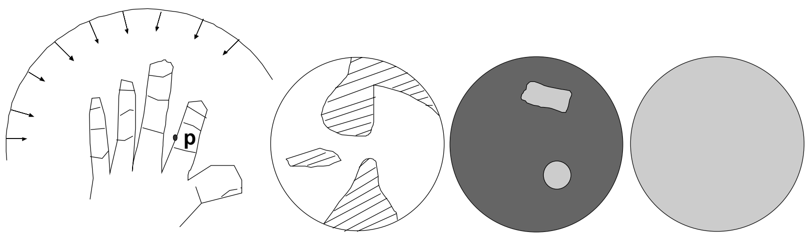


Figure 9.1: Rendering as Convolution of Lighting, Visibility, and BRDF

Considering only the direct contribution from the environment lighting, the rendering equation is an integration of the environmental lighting function, visibility function, and reflectance function, as shown in Equation (9.1).

$$L_o(\vec{p}, w_o) = \int_{\Omega} L_i(w_i : \vec{n}(\vec{p})) \cdot f_r(\vec{p}, w_i, w_o) \cdot v(w_i : \vec{p}, \vec{n}(\vec{p})) \cos\theta_i dw_i \tag{9.1}$$

where, the visibility function $v(w_i : \vec{p}, \vec{n}(\vec{p}))$ is a discontinuous binary function, while the lighting $L_i(w_i : \vec{n}(\vec{p}))$ and the BRDF $f_r(\vec{p}, w_i, w_i)$ are real functions. Figure 9.1 illustrates the scenario of the rendering and its three components.

The visibility function can be represented as either cube map or spherical harmonics coefficients. Cube map is one convenient way to store the functions defined over hemisphere and sphere. Cube map representations of the visibility function and other integrand functions in Equation (9.1) convert the integration of Equation (9.1) into a product sum of the integrand functions.

Spherical harmonics can be applied to transform the integration of two spherical functions into a vector dot product. We can use this property of spherical harmonics for fast evaluation of Equation (9.1). We combine L_i, v, and $\cos\theta_i$ into one group, and project this group into spherical harmonics basis and compute l_m for each basis function b_m, $m=0,1,\ldots$, using Equation (9.2).

$$l_m = \int_{\Omega} \left(L_i \cdot v \cdot \cos\theta_i\right) \cdot b_m dw \tag{9.2}$$

Equation (9.2) incorporates the visibility function into the SH coefficients. Visibility function together with low frequency environmental lighting L_i makes a low frequency function and hence can be approximated as a few SH coefficients.

Table 9.1: Pseudocode of Visibility Caching Algorithm

Procedure A: Interpolation

```
search cache to find available samples
for each candidate in the available sample set
    compute its error using Equation (9.4)
    if error is bigger than some threshold
        throw away this sample
    if final candidate set not empty
        Interpolate using Equation (9.3)
    else
        compute new sample
```

Procedure B: compute new sample

```
For each other scene elements
    Project to the visibility plane
    "or" with its visibility
    use smaller depth to update the depth plane
    convert visibility by incident lighting to SH coefficients
    compute $R_i$ using depth value
```

Visibility function v for neighboring surface points is very similar. Using ideas similar to Ward's irradiance interpolation, the above l_m at point p can be interpolated using cached values, as shown in Equation (9.3).

$$l_m^{\bar{p}} = \frac{\sum_{\bar{p}_i \in S} 1/\varepsilon_i^{\bar{p}} \cdot l_m^{\bar{p}_i}}{\sum_{\bar{p}_i \in S} 1/\varepsilon_i^{\bar{p}}} \tag{9.3}$$

where, $\vec{p}_i, p_i$ are the sampling point and its neighboring point, $\vec{R}_i$ is the average distance to the occluders. And,

$$\varepsilon_i(\vec{p}) = \frac{\left|\vec{p} - \vec{p}_i\right|}{R_i} + \sqrt{1 - \vec{N}(\vec{p}) \cdot \vec{N}(\vec{p}_i)} \qquad (9.4)$$

Such a caching procedure is described in pseudo-code shown in Table 9.1. As in Radiance, the samples can be stored and retrieved using an *octree* data structure. The interpolation accuracy can be improved by introducing spherical harmonics coefficient gradients with respect to normal and position changes [Annen 2004].

9.2.2 Perceptual based HDR Image Encoding

HDR images are researched almost exclusively in the computer graphics field in the past two decades. An effective HDR image compression will play a key role in the widespread use of HDR images in many applications. Such an argument is convincing if we think of the role that conventional image compression methods (JPEG, GIF, PNG) played in the widespread application of LDR images.

Bit rate is the average number of bits per pixel in a compressed image. It is used to measure the compression efficiency of the digital imaging encoding methods. Lowest bit rate is desirable for the same visual quality of compressed image. There are two ways to gain minimum bit rate: removing statistical redundancy and removing perceptual irrelevant information. Both have been exploited in developing LDR image compression standards, e.g., JPEG, GIF, JPEG2000. JPEG2000 has a more advanced application of the HVS properties and a more advanced entropy encoder than JPEG.

Research on HDR image compression has become very active in recent years. Besides the general purpose lossless compression algorithms, e.g. RLE, PIZ, perceptually based compression methods are essential to achieve aggressive compression of HDR images. Several research efforts in this direction have been reported in recent years [Mantiuk 2004; Li 2005]. However, to our knowledge, no one has tried to apply the CSF and the visual masking, which have been well exploited in LDR image compression, to the HDR image compression. In our opinion, it is time to fill this gap.

As discussed in the beginning of this chapter, a HDR image compression standard will be the ultimate goal of HDR image encoding research. But, none of the existing HDR image compression methods qualifies to be a HDR image compression standard, because these methods have not matured yet. Besides, an encoding standard needs to consider not only high compression ratio, but also additional requirements from various HDR image applications.

We can formulate the perceptual HDR image compression problem into two issues: given the desired bit rate, compute the compressed image that maximizes its visual fidelity to the given source image under given viewing/displaying conditions (Perceptually most efficient); given an image, compute the compressed image that uses the minimum possible bit rate to keep its visual identity to the given image under given viewing/displaying condition (Perceptual lossless).

The separation of perceptually irrelevant information is inspired by the first steps in the seeing process of the HVS. The light first falls onto the photoreceptors of the human eyes, and a non-linear (can be conveniently approximated by a log operation) processing is applied before the signal reaches the bipolar and ganglion cells, where an opponent visual signal is generated (can be approximated by a transformation to YCbCr color space). The opponent visual signal finally arrives at the visual cortex via the lateral geniculate nuclei (LGN), where contrast adaptation (CSF, visual masking) takes place (can be approximated by visual frequency weighting, self-contrast masking, neighboring masking) in multi-scale mechanisms.

Following a similar process, it is possible to first map the RGB information to the logarithm domain or a more accurate domain (logarithm for photopic conditions and power function for mesopic and scotopic conditions); transform this output to the YC_bC_r domain; transform the output from the YC_bC_r domain to a series of sub-bands, applying CSF weighting and pixel-wise non-linearity on the coefficients of all high pass bands; and finally, uniformly quantize the result in preparation for entropy encoding. When the uniform quantization step sizes are one, the compression is visually lossless. The issues are to decide the visual weights and the point-wise non-linearity.

Visually insignificant information is hidden from view through various mechanisms that have been modeled using threshold verses intensity (TVI) function, contrast sensitivity function (CSF) and visual masking function. So, the visually insignificant information should be first separated and discarded.

Some of the image compression research has succeeded in discarding part of the visually unimportant information. Mantiuk's uses TVI function in luminance domain to provide non-uniform quantization of the luminance channel of HDR images. JPEG uses CSF to apply non uniform quantization to the coefficients in DCT blocks of LDR images. JPEG2000 uses CSF and visual masking over the coefficients of wavelet tiles of LDR images.

JPEG2000 standard [ISO/IEC 2001] applies self-contrast masking and neighborhood masking to the wavelet coefficients x_i to get z_i, which are uniformly quantized further, as shown in Equation (9.5).

$$y_i = \frac{sign(x_i)|x_i|^{\alpha}}{w_i} = \frac{sign(x_i)|x_i|^{\alpha}}{1+\left(a\sum_{k\in neighborhood}|\hat{x}_k|^{\beta}\right)/|\phi_i|} \tag{9.5}$$

where,

$$a = \left(10000/2^{component\_bit\_depth-1}\right)^{\beta}$$
$$\alpha = 0.7; \beta = 0.2; N = 5; |\phi_i| = 12$$

Li [2005] proposes an interesting "companding" technique to range compress a 12-bit image to an 8-bit image by multiplying the gain map with the multi-resolution bands, and to recover the 12-bit image from its 8-bit version by dividing the 8 bit image gain map from the 8-bit image multi-resolution bands. The 8-bit image is then data compressed using the standard LDR image compression method. Log encoding is used to model the adaptation of the photoreceptors, and gamma-like mapping is used to derive a gain map in [Li 2005]. The gamma-like function is shown as Equation (9.6), and it is in similar to the Equation (9.5), because both use a pixel-wise non-linearity in the form $x' = \frac{x^{\alpha}}{g(N(x))}$.

$$G_i(x,y) = \left(\frac{A_i(x,y)+\varepsilon}{\delta}\right)^{\gamma-1} \tag{9.6}$$

where,

$$\delta = \alpha_i \sum_{(x,y)} A_i(x,y) \Big/ M \times N$$

The design of an HDR image encoding format involves the issues of pixel coding and image coding. Pixel coding considers the internal color representation (color space, data type), and image coding considers luminance coding (perceptual luminance information loss), chrominance coding (chrominance information loss). Figure 9.2 shows an encoding scheme that may take advantage of CSF and visual masking for perceptual coding in wavelet domain.

The compression scheme in Figure 9.2 lead to further issues: find the weights due to CSF effects of viewing HDR images; find the mapping functions for self masking and neighborhood masking due to the visual masking effects of viewing HDR images.

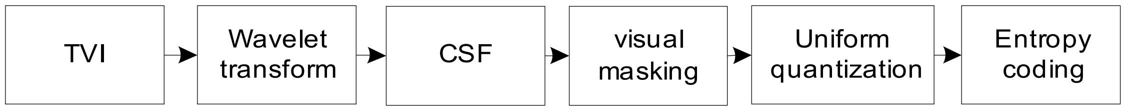


Figure 9.2: Wavelet Based Scheme for Perceptual HDR Images Encoding

An alternative way of perceptual HDR image encoding is to make use of a noticeable difference (JND) map: derive the threshold/supra-threshold map due to multiple HVS effects, and then use it to drive a rate-distortion process.

9.2.3 Lossless Data Encoding with Minimum Bits

Many kinds of data in computer graphics (HDR images, terrains, textures) use 32 bits floats to satisfy their precision requirement. Since most data have a smaller dynamic range or lower precision requirement than those provided by floating number representation, some bits are wasted in denoting redundant high dynamic range or precision; thus there is a wastage of storage space, even when *half* (16 bits floats) is used. Adaptive log encoding is a practical approach to a more efficient data representation. This research proposes a simple method to determine the minimum number of bits required in adaptive log encoding in order to satisfy desired error bounds. The method can thus provide the desired accuracy in terms of relative error with minimum number of bits, and thus will provide aggressive lossless encoding.

This proposal is inspired by Ward's 32 bits log encoding of HDR images [1998], wherein each pixel is first transformed into CIE *Lu'v'* color space, and then the luminance *L* is sent for log encoding as shown in Equation (9.7).

$$\begin{aligned} L' &= \log_{10}(L) \\ L'' &= \lceil 256(L'+64) \rceil \end{aligned} \tag{9.7}$$

This encoding is able to provide up to 38 orders of magnitude and keep the relative error below 0.3%. However, this encoding suffers from two problems. First, the relative error is fixed, therefore smaller relative error, though desirable in some applications, cannot be achieved. Second, 38 orders of magnitude is not always fully used in practical application; therefore the most significant bits are often wasted in encoding middle/low dynamic range signals.

Ward's encoding approach can be improved by introducing the adaptive encoding method, as shown in Equation (9.8).

$$\begin{aligned} L' &= \log_{10}(L) \\ L'' &= \left\lfloor \frac{L' - L'_{\min}}{L'_{\max} - L'_{\min}} (2^n - 1) \right\rceil \end{aligned} \tag{9.8}$$

where $L^{'}_{min}$ and $L^{'}_{max}$ are respectively the minimum log pixel value (other than zero) and the maximum log pixel value, *n* is the number of bits The optimal value of n is determined using the method described below. Denote the dynamic range as

$$D = L_{\max} - L_{\min} \, .$$

With the desired relative error known beforehand, the adaptive encoding will provide a means to encode any signal with a minimum number of bits, which is closely related to the final data size.

The quantization error in the logarithm domain will be half of the quantization step size Δ shown in Equation (9.9).

$$\Delta = D/(2^n - 1) \tag{9.9}$$

The relative error E can be computed as

$$E = 10^{D/(2^n-1)} - 1\,.$$

By expanding the power term in the above equation using Taylor series and keeping the first term, we get Equation (9.10), which approximates the relative error.

$$E \approx 2.3D/(2^n - 1) \tag{9.10}$$

Table 9.2 shows errors corresponding to a list of dynamic range and precision pairs. In the table the column entitled E is the corresponding error for the dynamic range shown in the column entitled D and number of bits shown in column n. The third row shows that, for a scene with typical dynamic range 4, the visual threshold 1% can be obtained with minimum 10 bits.

Table 9.2: Error List of Our Adaptive Data Encoding

D	n	E
2	8	1.8%
4	8	3.7%
4	10	0.93%
4	13	0.11%
10	15	0.07%
14	15	0.1%
29	16	0.1%
30	16	0.11%
76	16	0.27%

The minimum number of bits *n* that satisfy the desired relative error *E* for dynamic range *D* is obtained by taking the logarithm base 2 of both sides of Equation (9.10) and rearranging the terms.

$$n = \lceil \log_2 D - \log_2 E + 1.2 \rceil \qquad (9.11)$$

We present two examples to verify our method to calculate the minimum number of bits. For a typical single HDR image with dynamic range *D*=4 and tolerated error *E*=1%, the minimum number of bits is calculated as 2+6.64+1.2=9.84, so *n*=10. Conversely, when *D*=4 and *n*=10, the error is $10^{2*(4/(210+1-2))}$-1=0.93%, which is approximately 1%. Taking another example, when *D*=10 and *E*=0.1%, the minimum number of bits is 3.323+9.97+1.2=14.49, so *n*=15. Conversely, when *D*=10 and *n*=15, the error is $10^{2*(10/215+1-2)}$-1=0.07%, which is less than 0.1%.

Compared to the *half* data type used in today's graphics hardware and OpenEXR [ILM 2004], our new encoding method excels in several aspects. First, our method can encompass 14 orders of magnitude at 0.1% relative error with 16 bits (15 magnitude bits plus 1 additional sign bit) (see Table 9.2). Our method can save bits when encoding data of smaller dynamic range. Second, our method doesn't suffer from the *denormalization* problem in OpenEXR [ILM 2004]. Finally, as shown in Equation (9.8), our method encodes real values with integer numbers, which lends itself to better compression than floating point numbers.

One direct practical application of our adaptive encoding is to improve Ward's 32 bits *LogLuv* pixel format by using less bits to encode the magnitude of the *LogL* channel with our method.

Given a raw HDR image, first convert it to the perceptual uniform CIE $Lu'v'$ color space as Ward's 32 bits *LogLuv* pixel format does, then transform the luminance channel into the logarithm domain and find its dynamic range, use it to compute the number of bits to encode the luminance channel using Equation (9.11), and then quantize the luminance channel in the logarithm domain using Equation (9.8) and quantize the chrominance channels as Ward's 32bits *LogLuv* pixel format does, finally sending all channels to an image entropy compressor.

LIST OF REFERENCES

[1] Adams, M.D. and Kosentini, F., JasPer: A Software-based JPEG-2000 Codec Implementation. In *Proceedings of IEEE International Conference on Image Processing,* Vancouver, BC, Canada, October, pages 53-56. 2000.

[2] Alliez, P., Meyer, M., and Desbrun, M., Interactive Geometry Remeshing. In *Proceedings of ACM SIGGRAPH 2002*, pages 347-354.

[3] Annen, T., Kautz, J., Durand, F., and Seidel, H, -P., Spherical Harmonics Gradients for Mid-Range Illumination. *In Proceedings of EGSR 2004,* pages 331-336.

[4] Arvo, J. and Kirk, D., Particle Transport and Image Synthesis. In *Computer Graphics* (SIGGRAPH 1990 Proceedings), pages 63-66.

[5] Ashikhmin, M. A., Tone Mapping Algorithm for High Contrast Images. In *13th Eurographics Workshop on Rendering (2002)*, pages 1-11.

[6] Bala, K., *Radiance Interpolants for Interactive Scene Editing and Ray Tracing.* Ph.D. diss., MIT., 1999a.

[7] Bala, K, Dorsey, J., and Teller, S., Radiance Interpolants for Accelerated Bounded-Error Ray Tracing. *ACM Transactions on Graphics* 18(3), pages 213-256. 1999b.

[8] Bala, K, Dorsey, J. and Teller, S., Interactive Ray-Traced Scene Editing Using Ray Segment Trees. In 10th *Eurographics Workshop on Rendering*, pages 39-52. 1999c.

[9] Bala, K., Walter, B. J., and Greenberg, D. P., Combining Edges and Points for Interactive High-Quality Rendering. In *ACM Transactions on Graphics* (Proceedings of ACM SIGGRAPH 2003), 22(3), pages 631-640, 2003.

[10] Barash, D., Bilateral Filtering and Anisotropic Diffusion: Towards a Unified Viewpoint. In *Third International Conference on ScaleSpace and Morphology* (2001), pages 273-280.

[11] Blinn. J., Models of Light Reflection For Computer Synthesized Pictures. In *Proceedings of SIGGRAPH 1977*, pages 192–198.

[12] Bolin, M.R., and Meyer, G.W., A Perceptually Based Adaptive Sampling Algorithm. In *Proceedings of SIGGRAPH 1998*, pages 299-310.

[13] Catmull, E., *A Subdivision Algorithm for Computer Display of Curved Surfaces*, University of Utah, Salt Lake City, December, 1974.

[14] Chen, S.E., Rushmeier, H.E., Miller, G., and Turner, D., A Progressive Multi-Pass Method for Global Illumination. in *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques*, pages 165-174. 1991.

[15] Chen, W. –C., Bouguet, J. –Y., Chu, M., and Grzeszczuk, R., Light Field Mapping: Efficient Representation and Hardware Rendering of Surface Light Field. In *Proceedings of ACM SIGGRAPH 2002*, pages 447-456.

[16] Cohen, J., Tchou, C., Hawkins, T., and Debevec, P., Real-time high dynamic range texture mapping. *Proceedings of 12th Eurographics Workshop on Rendering*, pages 313-320. 2002.

[17] Cook. R. and Torrance, K. A., Reflectance Model for Computer Graphics. In *Proceedings of SIGGRAPH 1981*, pages 307–316.

[18] Dashti, A.E., Kim, S.H., Shahabi, C., and Zimmermann, R., *Streaming Media Server Design*, Prentice Hall, 2003.

[19] Debevec, P.E., and Malik, J., Recovering High Dynamic Range Radiance Maps from Photographs. In *Proceedings of SIGGRAPH 1997,* pages 369-378.

[20] Debevec P., Rendering Synthetic Objects into Real Scenes: Bridging Traditional And Image-based Graphics with Global Illumination and High Dynamic Range Photography. In *Proceedings of ACM SIGGRAPH 1998*, M. Cohen, Ed., pages 189-198.

[21] DiCarlo, J. M., and Wandell, B. A., Rendering high dynamic range images. In *Proceedings of the SPIE: Image sensors, 3965* (2001), pages 392-401.

[22] Dmitriev, K., Brabec, S., Myszkowski, K., and Seidel, H. -P., Interactive Global Illumination Using Selective Photon Tracing. In *Proceeding of 13th Eurographics Workshop on Rendering*, pages 25-36. 2002.

[23] Drettakis, G. and Sillion, F. X., Interactive Update of Global Illumination Using a Line-Space Hierarchy. In *Proceedings of SIGGRAPH 1997,* pages 57-62.

[24] Durand, F. and Dorsey, J., Fast Bilateral Filtering for the Display of High-Dynamic-Range Images. In *Proceedings of ACM SIGGRAPH 2002,* pages 257-266.

[25] Elad, M., Analysis of the Bilateral Filter. In *The 36th Asilomar on Signals, Systems and Computers,* Pacific Grove, CA, 2002a.

[26] Elad, M., On the Origin of the Bilateral Filter and Ways to Improve It. In *IEEE Trans. On Image Processing,* 11(10), pages 1141-1151. 2002b.

[27] Fattal, R., Lischinski, D., and Werman, M., Gradient Domain High Dynamic Range Compression. In *Proceedings of SIGGRAPH 2002,* pages 249-256.

[28] FFMPEG. http://ffmpeg.sourceforge.net/index.org2.html. 2004.

[29] Fleishman, S., Drori, I., and Cohen-Or, D., Bilateral Mesh Filtering. In *Proceedings of ACM SIGGRAPH 2003*, pages 950-953.

[30] Funkhouser, T. A. and Carlo, H. S., Adaptive Display Algorithm for Interactive Frame Rates During Visualization of Complex Virtual Environments. In *Proceedings of SIGGRAPH 1993*, pages 247-254.

[31] Gibson, J. D., Berger, T., Lookabaugh, D., Lindbergh, D., and Baker, R. L., *Digital Compression for Multimedia*. Morgan Kaufmann Publishers, Inc., 1998.

[32] Goral, C.M., Torrance, K.E., Greenberg, D.P., and Battaile B., Modeling the Interaction of Light between Diffuse Surfaces. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, pages 213-222. 1984.

[33] Gortler, S., Grzeszczuk, R., Szeliski, R., and Cohen, M.F., The Lumigraph. In *Proceedings of SIGGRAPH 1996*, pages 43-54.

[34] Greenberg, D., Cohen, M., and Torrance, K., Radiosity: A Method for Computing Global Illumination. *Visual Computing,* 2(5), pages 291-297. 1986.

[35] Greenberg, D., Foo, S.-C., Torrance, K.E., Shirley, P., Arvo, J., Lafortune, E., Ferwerda, J.A., Walter, B., Trumbore, B., and Pattanaik, S., A Framework for Realistic Image Synthesis. In *ACM SIGGRAPH 1997*, pages 477-494.

[36] Greger, G., Shirley, P., Hubbarad, P. M., and Greenberg, D. P., The Irradiance Volume. In *IEEE Computer Graphics and Applications*, 18(2), pages 32-43. 1998.

[37] Gu, X., Gortler, S. J., and Hoppe, H., Geometry Images. In *Proceedings of ACM SIGGRAPH 2002*, pages 355-361.

[38] Heidrich, W., Daubert, K., Kautz, J., and Seidel, H.-P., Illuminating Micro Geometry Based on Precomputed Visibility. In *Proceedings SIGGRAPH 2000*, pages 455-464.

[39] Hunt, R.W.G., *The Reproduction of Color*. Fountain Press, England, 1995.

[40] ILM. http://www.openexr.org/. 2004.

[41] International Organization for Standardization and International Electrotechnical Commission, *ISO/IEC 15444-1: 2000*, *Information Technology-JPEG2000 image coding system-Part 1: Core Coding System*. 2000.

[42] International Organization for Standardization and International Electrotechnical Commission, *ISO/IEC 15444-2: 2000*, *Information Technology-JPEG2000 image coding system-Part 2: Extensions*. 2001.

[43] James, D.L. and Fatahalian, K., Precomputing Interactive Dynamic Deformable Scenes. In *ACM Transactions on Graphics* (Proceedings of ACM SIGGRAPH 2003), 22(3), pages 879-887. 2003.

[44] JasPer. http://www.ece.uvic.ca/~mdadams/jasper/index.html. 2004.

[45] Jensen, H.W. and Christensen, N.J., Optimizing Path Tracing using Noise Reduction Filters. In *Proceedings of the WSCG 1995*, pages 134-142.

[46] Jensen, H.W., Global Illumination using Photon Maps. In *Rendering Techniques1996*, Springer-Verlag, pages 21-30.

[47] Jones, T. R., Durand, F., and Desbrun, M., Non-Iterative, Feature-Preserving Mesh Smoothing. In *Proceedings of ACM SIGGRAPH 2003*, pages 943-949.

[48] Kajiya, J., The Rendering Equation. In *Proceedings of SIGGRAPH 1986*, pages 143-150.

[49] Kautz, J., Lehtinen, J., and Aila T., Hemispherical Rasterization for Self-Shadowing of Dynamic Objects. In *Proceedings of Eurographics Symposium on Rendering 2004*, pages 179-184.

[50] Keller, A., Instant Radiosity. in *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, pages 29-56. 1997.

[51] Khodakovsky, A., Litke, N., and Schroder, P., Globally Smooth Parameterizations with Low Distortion. In *Proceedings of ACM SIGGRAPH 2003*, pages 350-357.

[52] Kollig, T. and Keller, A., Efficient illumination by high dynamic range images. *Proceedings of the 13th Eurographics workshop on Rendering*, pages 45-50. 2003.

[53] Kovar, L., Gleicher, M., and Pighin, F., Motion Graphs. In *Proceeding of ACM SIGGRAPH 2002*, pages 473-482.

[54] Lafortune, E. P., Willems, Y. D., Bi-directional Path Tracing. In *Proceedings of 3rd International Conference on Computational Graphics and Visualization Techniques (Compugraphics'93)*, pages 145-153. 1993.

[55] Lafortune, E.P. and Willems, Y.D., A Theoretical Framework for Physically Based Rendering. *Computer Graphics Forum,* 13(2), pages 97-107. 1994.

[56] Lafortune, E. P., Foo, S., Torrance, K., and Greenberg, D., Non-Linear Approximation of Reflectance Functions. In *Computer Graphics Proceedings*, Annual Conference Series, ACM SIGGRAPH, pages 117-126. 1997.

[57] Larson, G.W., Real pixels. *Graphics Gems II*, pages 80-83, Academic Press, 1991.

[58] Larson, G.W., LogLuv encoding for full-gamut, high-dynamic range images. In *Journal of Graphics Tools*, *3* (1998), pages 15-31.

[59] Lee, M.E. and Redner, R.A., Filtering: A note on the Use of Nonlinear Filtering in Computer Graphics. In *Computer Graphics* 10(3) (May/June 1990), pages 23-29.

[60] Lehtinen, J. and Kautz, J., Matrix Radiance Transfer. In *Proceedings of ACM SIGGRAPH 2003 Symposium on Interactive 3D Graphics*, pages 59-64.

[61] Levoy, M. and Hanrahan, P., Light Field Rendering. In *Proceedings of SIGGRAPH 1996,* pages 31-42.

[62] Li, Y.Z., Sharan, L., and Adelson, E.H., Compressing and Companding High Dynamic Range Images with Subband Architectures. In *Proceedings of SIGGRAPH 2005,* pages 836-844.

[63] Lubin, J. A., Visual Discrimination Model for Imaging System Design and Evaluation. In *Visual Models for Target Detection and Recognition*, World Scientific Publishers, Peli, E. (ed.), pages 245-283. 1995.

[64] Luebke, D. and Erikson, C., View-Dependent Simplification of Arbitrary Polygonal Environments. In *Proceedings of SIGGRAPH 1997*, pages 199-208.

[65] Mallat, S. G., A Theory for Multiresolution Signal Decomposition: The Wavelet Representation. *IEEE Trans. Pat Anal Mach Intell, 11*(July 1989), pages 676-693.

[66] Mantiuk, R., Krawczyk, G., Myszkowski, K., and Seidel, H.-P., Perception-Motivated High-Dynamic-Range Video Encoding. In *Proceedings of ACM SIGGRAPH*, pages 733-741. 2004.

[67] Marpe, D., Blattermann, G., and Wiegand, T., Adaptive Codes for H.26L. *JVT Document JVT-L13*, Eibsee, January, 2001.

[68] McCool, M. D., Anisotropic Diffusion for Monte Carlo Noise Reduction. In *ACM Trans. On Graphics,* 18(2) (April, 1999), pages 171-194.

[69] Milgram, P. and Kishino, F., A Taxonomy of Mixed Reality Visual Displays, IEICE Transactions on Information Systems, vol.E77-D, no.12. 1994.

[70] Mitsunaga, T. and Nayar, S. K., Radiometric self calibration. in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2* (1999), pages 374-380.

[71] Nadenau, M. J. and Reichel, J., Compression of Color Images with Wavelets under Consideration of the HVS. in *Proceedings of IS&T/SPIE Conference on Human Vision and Electronic Imaging IV,3644*, San Jose, California, January 1999, pages 129-140.

[72] Nijasure, M., Pattanaik, S.N., and Goel, N., Interactive Global Illumination in Dynamic Environments Using Commodity Graphics Hardware. In *Proceedings of Pacific Conference on Computer Graphics and Applications 2003*, pages 450-454.

[73] Osher, S. and Sethian, J.A., Fronts propagating with Curvature-Dependent Speed: Algorithm Based on Hamilton-Jacobi Formulations. *Journal of Computational Physics 79*(1988), pages 12-49.

[74] Pannebaker, W.B. and Gall, D.L., *Still Image Data Compression Standard*. Van Nostrand Reinhold, New York, NY, 1993.

[75] Pannebaker, W.B., Gall, D.L., and Reinhold V.N., *MPEG Digital Video Compression Standard*. Van Nostrand Reinhold, 1995.

[76] Parker, S., Martin, W., Sloan, P.-P., Shirley, P., Smits, B., and Hansen, C., Interactive Ray Tracing. In *Interactive 3D Graphics*, pages 119-126. 1999.

[77] Pattanaik, S.N. and Mudur, S.P., Computation of Global Illumination by Monte Carlo Simulation of the Particle Model of Light. In *Proceedings of the Third Eurographics Workshop on Rendering*, pages 71-83. 1992.

[78] Pattanaik, S.N. and Mudur, S.P., The Potential Equation and Importance in Illumination Computations. *Computer Graphics Forum*, 12(2), pages 131-136. 1993.

[79] Pattanaik, S.N., Ferwerda, J.A., Fairchild, M.D., and Greenberg, D.P., A Multiscale Model of Adaptation and Spatial Vision for Realistic Image Display. In *Proceedings of SIGGRAPH 1998*, pages 287-298.

[80] Pattanaik, S.N., Tumblin, J.E., Yee, H., and Greenberg, D.P., Time-Dependent Visual Adaptation for Realistic Real-Time Image Display. In *Proceedings of SIGGRAPH 2000,* pages 47-53.

[81] Pattanaik, S.N., and Yee, H., Adaptive Gain Control for High Dynamic Range Image Display. In *Proceedings of Spring Conference in Computer Graphics (SCCG 2002)*, Budmerice, Slovak Republic (April, 2002), pages 24-27.

[82] Phong, B.-T., Illumination for Computer Generated Pictures. *Communications of the ACM*, 18(6) pages 311–317. 1975.

[83] Poynton, C.A., *A technical Introduction to digital video*. John Wiley & Sons, Inc., 1996.

[84] Poynton, C., *Digital video and HDTV algorithms and interfaces*. Morgan Kaufmann Publishers, 2003.

[85] Purcell, T.J., Buck, I., Mark, W.R., and Hanrahan, P., Ray Tracing on Programmable Graphics Hardware. In *Transactions on Graphics*, 21(3), pages 703-712. 2002.

[86] Purcell, T.J., Donner, C., Cammarano, M., Jensen, H.W., and Hanrahan, P., Photon Mapping on Programmable Graphics Hardware. in *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware*, pages 41-50. 2003.

[87] Purgathofer, W., A Statistical Method for Adaptive Stochastic Sampling. In *Computers and Graphics*, 2(2) (1987), pages 157-162.

[88] Rabbani, M. and Joshi, R. An Overview of the JPEG2000 Still Image Compression Standard. In *Signal Processing: Image Communication*, 17(3) (2002), pages 3-48.

[89] Rahman, Z., Jobson, D.J., and Woodell, G.A., Multiscale Retinex for Color Rendition and Dynamic Range Compression. In *SPIE International Symposium on Optical Science, Engineering, and Instrumentation, Conference on Signal and Image Processing XIX, 2847* (1996).

[90] Ramamoorthi, R. and Hanrahan, P., An Efficient Representation for Irradiance Environment Maps. In *Proceedings of ACM SIGGRAPH 2001*, pages 497-500.

[91] Ramamoorthi, R., Hanrahan, P., Frequency Space Environment Map Rendering. In *Proceedings of SIGGRAPH 2002*, pages 517-526.

[92] Ramasubramanian, M., Pattanaik, S.N., and Greenberg, D.P., A Perceptually Based Physical Error Metric for Realistic Image Synthesis. In Proceedings of ACM SIGGRAPH 1999, pages 73-82.

[93] Reinhard, E., Stark, M., Shirley, P., and Ferwerda, J., Photographic Tone Reproduction for Digital Images. In *Proceedings of SIGGRAPH 2002*, pages 267-276.

[94] Richardson, I.E.G., *H.264 and MPEG-4 video compression: Video Coding for Next-generation Multimedia*. John Wiley & Sons, Inc., 2003.

[95] Rosenholtz, R. and Watson, A.B., Perceptual Adaptive JPEG Coding. in *Proceedings of IEEE International Conference on Image Processing 1*, pages 901-904, Lausanne, Switzerland, 1996.

[96] Rubinstein, R.Y., *Simulation and the Monte Carlo Method*. John Wiley & Sons, 1981.

[97] Rushmeier, H. E. and Ward, G. J., Energy Preserving Non-linear filters. In *Proceedings of ACM SIGGRAPH 1994,* pages 131-138.

[98] Sadka, A.H., *Compressed video communications*. John Wiley & Sons, Inc., 2002.

[99] Salomen, D., *Data Compression: The complete Reference* (2nd edition), Springer, 2000.

[100] Sander, P.V., Gortler, S.J., Snyder, J., and Hoppe, H., Signal-Specialized Parameterization. In *Thirteenth Eurographics Workshop on Rendering* (2002), pages 87-98.

[101] Schlick, C., Quantization techniques for visualization of high dynamic range pictures. In *Proceedings of 5th Eurographics Workshop* (June 1994), pages 7-20.

[102] SGI. http://positron.cs.berkeley.edu/~gwlarson/pixformat/. 1997.

[103] Shirley, P., Wang, C., and Simmerman, K., Monte Carlo methods for direct lighting calculations. In *ACM Transactions on Graphics 15*, 1 (1996), pages 1-36.

[104] Simoncelli, E.P., and Adelson, E.H., Noise Removal via Bayesian Wavelet Coring. In *Proceedings of 3rd IEEE Int'l Conference on Image Processing*, Lausanne, Switzerland, pages 379-382, September, 1996.

[105] Simoncelli, E.P., Bayesian Denoising of Visual Images in the Wavelet Domain. In *Bayesian Inference in Wavelet Based Models*, eds Muller P. and Vidakovic, B., pages 291-308, Springer-Verlag, New York, 1999.

[106] Simmons, M. and Sequin, C.H., Tapestry: A Dynamic Mesh-Based Display Representation for Interactive Rendering. In *7th Eurographics Workshop on Rendering,* pages 329-340. 2000.

[107] Simoncelli, E.P., LCV. http://www.cns.nyu.edu/~eero/software.html, 2004.

[108] Sloan, P.-P., Kautz, J., and Snyder, J., Pre-computed Radiance Transfer for Real-time Rendering in Dynamic, Low-frequency Lighting Environments. In *ACM Transactions on Graphics,* 21(3) (2002), pages 527-536.

[109] Sloan, P.-P., Liu, X.H., Shum, -Y. and Synder, J., Bi-Scale Radiance Transfer. In *ACM Transactions on Graphics* (Proceedings of ACM SIGGRAPH 2003), 22(3), pages 370-375. 2003a.

[110] Sloan, P.-P., Hall, J., Hart, J., and Snyder, J., Clustered Principal Components for Pre-computed Radiance Transfer. In *ACM Transactions on Graphics* (Proceedings of ACM SIGGRAPH 2003), 22(3), pages 382-391. 2003b.

[111] Strutz T., Adaptive Quantization for Lossy Image Compression Controlled by Noise Detection. In *Proceedings of DCC'2001*, Snowbird, Utah, USA, pages 27-29.

[112] Taubman, D.S. and Marcellin, M.W., *JPEG2000: Image Compression Fundamentals, Standards, and Practice*. Kluwer Academic Publishers, Boston, 2002.

[113] Tole, P., Pellacini, F., Walter, B., and Greenberg, D.P., Interactive Global Illumination in Dynamic Scenes. In *ACM Transactions on Graphics* (Proceedings of ACM SIGGRAPH 2002), 21(3), pages 537-546. 2002.

[114] Tomasi, C. and Manduchi, R., Bilateral Filtering for Gray and Color Images. In *Proceedings of the IEEE ICCV 1998*, pages 839-846, Bombay, India.

[115] Tamstorf, R. and Jensen, H.W., Adaptive Sampling and Bias Estimation in Path Tracing. In *Rendering Techniques '97*. Eds. Dorsey, J. and Slusallek, P.H., Springer-Verlag, pages 285-295.

[116] Teller, S, Bala, K., and Dorsey, J., Conservative Radiance Interpolants for Ray Tracing. In *7th Eurographics Workshop on Rendering*, pages 258-269. 1996.

[117] Tumblin, J., Hodgins, J. K., and Guenter, B.K., Two methods for display of high contrast images. In *ACM Transactions on Graphics,* 18(1), pages 56-94. 1999a.

[118] Tumblin, J., and Turk, G., LCIS: A boundary hierarchy for detail-preserving contrast reduction. In *Proceedings of SIGGRAPH 1999*, pages 83-90. 1999b.

[119] Uchiyama, S., Takemoto, K., Satoh, K., Yamamoto, H., and Tamura, H., MR Platform: A Basic Body on Which Mixed Reality Applications Are Built. In *Proceedings of ISMAR'02*, pages 246-256. 2002.

[120] Veach, E. and Guibas, L. J., Metropolis Light Transport. In *ACM SIGGRAPH 1997 Proceedings* (Aug. 1997), T. Whitted, Ed., pages 65-76. 1997.

[121] Wald, I., Slusallek, P., Benthin, C., and Wagner, M., Interactive Rendering with Coherent Ray Tracing. In *Proceedings of Eurographics*, pages 153-164. 2001a.

[122] Wald, I. and Slusallek, P., State of Art in Interactive Ray Tracing. In *State of the Art Reports, Eurographics 2001*, pages 21-42. 2001b.

[123] Wald, I., Kollig, T., Benthin, C., Keller, A., and Slusallek, P., Interactive Global Illumination using Fast Ray Tracing. In *13th Eurographics Workshop on Rendering,* 2002.

[124] Wald, I., Benthin, C., and Slusallek, P., Distributed Interactive Ray Tracing of Dynamic Scenes. In *Proceedings of the IEEE Symposium on Parallel and Large-Data Visualization and Graphics* (PVG), pages 77-85. 2003.

[125] Wallace, J.R., Cohen, M.F., and Greenberg, D.P., A Two-Pass Solution to the Rendering Equation: A Synthesis of Ray Tracing and Radiosity Methods. *ACM SIGGRAPH 1987*, pages 311-320.

[126] Walter, B., Hubbard, P.M., and Parker, S., Interactive Rendering Using the Render Cache. In *10th Eurographics Workshop on Rendering.* 1999.

[127] Walter, B., Drettakis, G., and Greenberg, D.P., Enhancing and Optimizing the Render Cache. In *13th Eurographics Workshop on Rendering.* 2002.

[128] Wang, J., DWTGPU. http://www.cse.cuhk.edu.hk/~ttwong/demo/dwtgpu/dwtgpu.html. 2004.

[129] Ward, G.J., Rubinstein, F.M., and Clear, R.D., A Ray Tracing Solution for Diffuse Inter-reflection. *Computer Graphics,* 22(4) (August 1988), pages 85-92.

[130] Ward, G., Measuring and Modeling Anisotropic Reflection. In *Proceedings SIGGRAPH*, pages 265–272. 1992.

[131] Ward, G.L., The RADIANCE Lighting Simulation and Rendering System. In *Computer Graphics* (Proceedings of SIGGRAPH 1994 conference), pages 459-472. July 1994.

[132] Ward, G., Radiance E-mail Archive. http://radsite.lbl.gov/radiance/digests_html/v3n0.html. 1996.

[133] Ward, G.L., Rushmeier, H., and Piatko, C., A Visibility Matching Tone Reproduction Operator for High Dynamic Range Scenes. In *IEEE Transactions on Visualization and Computer Graphics*, 3(4) (1997), pages 291-306.

[134] Ward, G.L., *Rendering with Radiance: A Practical Tool for Global Illumination*. ACM SIGGRAPH'98 Course #33, July, 1998.

[135] Ward, G. and Simmons, M., The Holodeck Ray Cache: An Interactive Rendering System for Global Illumination. *ACM Transactions on Graphics*. 18(4), pages 361-398. 1999.

[136] Ward, G., HDR Encoding. http://www.anyhere.com/gward/hdrenc/hdr_encodings.html. 2004a.

[137] Ward, G. and Simmons, M., Subband Encoding of High Dynamic Range Imagery. In *APGV'04: Proceedings of the 1st Symposium on Applied Perception in Graphics and Visualization*, ACM Press, 2004, pages 83-90. 2004b.

[138] Ward, G., RADIANCE. http://radsite.lbl.gov/. 2005.

[139] Whitted, T., An Improved Illumination Model for Shaded Display. *Comm. ACM*, 23(6), pages 343-349. 1980.

[140] Xu, R. and Pattanaik, S.N., High Dynamic Range Image Display Using Level Set Framework. *Journal of WSCG,* 1-3(11). 2003.

[141] Xu, R., Pattanaik, S.N., and Hughes, C.E., Real-time Rendering of Dynamic Objects in Dynamic, Low-frequency Lighting Environments. In *Proceedings of CASA'2004*.

[142] Xu, R. and Pattanaik, S.N., Non-Iterative, Robust Monte Carlo Noise Reduction. *IEEE CG&A*, 25(2), pages 31-35. 2005a.

[143] Xu, R. and Pattanaik, S.N., Monte Carlo Noise Reduction Using Bayesian Method in Wavelet Domain. In *Proceedings of Second International Conference on Video, Vision and Graphics*, 2005b.

[144] Xu, R., Pattanaik, S.N., and Hughes, C.E., HDR Still Image in JPEG2000. *IEEE CG&A*. 2005c.

[145] Xu, R. and Pattanaik, S.N., Radiosity. *Wiley Encyclopedia of Electrical and Electronics Engineering Online*, J. Webster (ed.) (to appear). 2005d.

[146] Xu, R., Pattanaik, S.N., and Hughes, C.E., *Practical Sampling of 3D space for Realistic Rendering of Complex Scenes*. Technical Report. Computer Science, UCF. 2005e.

[147] Xu, R., Pattanaik, S.N., and Hughes, C.E., *High Dynamic Range Image and Video Data Compression*. Technical report. Computer Science, UCF. 2005f.