Models and Analyses for Image Synthesis

Cyril Soler

UNIVERSITÉ JOSEPH FOURIER - GRENOBLE

ÉCOLE DOCTORALE STIC
SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

HABILITATION à DIRIGER DES RECHERCHES

de l'Université de Grenoble
Mention : Informatique

Présentée et soutenue par

Cyril Soler

Models and Analyses for Image Synthesis

Préparée à l'INRIA Rhône Alpes,
Projet Maverick
soutenue le 24 Juin 2014

Jury :

Rapporteurs :	Bruno Levy	-	INRIA Lauraine, Nancy
	Mathias Paulin	-	Univerité Paul Sabatier, Toulouse
	Matthias Zwicker	-	Université de Berne, Suisse
Examinateurs :	Marie-Paule Cani	-	INPG, Grenoble
	Tamy Boubekeur	-	Telecom Paris Tech, Paris
	Pierre Poulin	-	Université de Montréal, Canada
Invité :	Wojciech Jarosz	-	Disney Research, Zurich, Suisse.

Remerciements

Je remercie en premier lieu les membres de mon jury qui ont eu la gentillesse de participer à cette entreprise.

Je tiens à remercier François Sillion et Nicolas Holzschuch pour m'avoir accueilli au sein des équipes ARTIS et Maverick, ainsi que le président de l'INRIA pour sa confiance.

Merci enfin à tous mes collaborateurs, thésards, et co-auteurs, assistantes, ainsi que les membres des équipes auxquelles j'ai appartenu, pour leur soutien, leur inspiration, et leurs conseils.

Merci à mes proches, à ma famille, et surtout à Claire.

Merci aux honnorables membres du CCCP pour considérer mon humble candidature.

Table of contents

1 Introduction **3**
- 1.1 Why research in computer graphics? . . . 3
- 1.2 My perception of research in computer graphics . . . 8
- 1.3 Position of the presented work . . . 11

2 Scientific activities **13**
- 2.1 Past PhD students . . . 13
- 2.2 Undergraduate students and engineers . . . 14
- 2.3 Funded projects . . . 15
- 2.4 Other collaborations . . . 17
- 2.5 Program committee and reviews . . . 17
- 2.6 Consulting at Digisens S.A. . . . 18
- 2.7 Code . . . 18

3 Research summary **21**
- 3.1 Simulation of radiative exchanges . . . 23
  - 3.1.1 The rendering equation . . . 23
  - 3.1.2 Radiative exchanges in plant models . . . 29
- 3.2 Fourier Analysis of Light Transport . . . 31
  - 3.2.1 The covariance representation . . . 33
  - 3.2.2 Application to depth of field and motion blur . . . 35
  - 3.2.3 Fourier analysis for density estimation methods . . . 37
  - 3.2.4 Fourier analysis of participating media . . . 38
- 3.3 Real time rendering . . . 41
  - 3.3.1 Revisiting birefringent materials . . . 42
  - 3.3.2 Screen-space indirect illumination . . . 44
  - 3.3.3 Rendering measured materials in real time . . . 45
- 3.4 Computational geometry . . . 48
  - 3.4.1 Seamless parameterization-free texturing . . . 48
  - 3.4.2 Automatic instantiation of geometry . . . 49
  - 3.4.3 Smoothing data for tomography reconstruction . . . 52
- 3.5 Computational imagery . . . 54
  - 3.5.1 Analysis and synthesis of stochastic distributions of sprites . . . 54
  - 3.5.2 Image segmentation from inaccurate input . . . 54
  - 3.5.3 Empirical mode image decomposition . . . 55

4 Research project **57**
- 4.1 Short term research . . . 59
- 4.2 Long term research . . . 61

A Publications **65**
A.1 Internationales journals with reviewing commitee 65
A.2 International conferences with reviewing commitee 66
A.3 Posters in international conferences 67
A.4 French journals 68
A.5 Communications in French conferences 68
A.6 Research reports 69
A.7 PhD Thesis 69

Bibliography **71**

B Key publications **83**

Résumé

La recherche en informatique graphique cherche à produire des algorithmes efficaces de génération et de manipulation de données, de simulation physique, et de visualisation temps-réel pour produire des images de synthèse. La croissance constante de la puissance des machines est talonnée par l'explosion de la demande en terme de complexité, de réalisme et d'interactivité, motivant la mise au point de méthodes toujours plus performantes pour générer des images de synthèse. Je présente dans ce contexte plusieurs contributions dans différents sous-domaines de l'informatique graphique: (1) l'analyse et le traitement de la géométrie, incluant un algorithme original de placage de texture ainsi qu'une méthode d'instantiation automatique de données géométriques; (2) le traîtement d'images avec une généralisation de la décomposition modale de Hilbert-Huang, et un algorithme de sélection semi-automatique de régions à partir d'indications imprécises; (3) la simulation de l'éclairage global et son application à la simulation physiologique de la croissance des plantes, l'analyse de Fourier de l'opérateur de transport de l'éclairage pour diverses applications, ainsi que des contributions en rendu temps-réel. Dans une seconde partie je détaille des pistes de recherche que je considère prometteuses.

Summary

Research in computer graphics aims at producing efficient algorithms for generating and manipulating data, simulating physical phenomena, and displaying information in real-time in order to generate synthetic images. The constant increase in computational power is challenged by the explosion of the demand in terms of level of realism, complexity of simulations and amount of data to handle in interactive applications. In this context, I present contributions in different sub-domains of research in computer graphics: (1) geometry analysis and processing including an original texture mapping algorithm and a method for automatic instancing of scenes; (2) image processing, for which I propose an extension of the Hilbert-Huang empirical mode decomposition in 2D and an algorithm for binary content selection from inaccurate user input; (3) simulation of light transport applied to plant growth simulation, and Fourier analysis of the light transport operator for various applications including real-time rendering. Eventually I present potential research tracks that I consider promising for the future.

Chapter 1

Introduction

"*As technology advances, rendering time remains constant*"

Jim Blinn

1.1 Why research in computer graphics?

Computers are ubiquitous, and because our primary interaction with computers is to look at their screen we need to find proper ways to generate and display information. Research in computer graphics is all about finding efficient algorithms to compute images. That includes designing appropriate data representations, processing this data, simulating the underlying physical phenomena, and being able to shape these algorithms for a specific kind of hardware. Of course, this is not as simple as it sounds, and research in computer graphics sometimes produces new theories at a more fundamental scope. But the ultimate goal is to produce images on computers to be viewed by humans. In order to understand what research is needed, it is required to understand what are the problems to solve, and what brings these problems on the table. A practical answer to this question is to look through the application domains and see what is the actual demand in computer-related techniques.

Movies. There's a strong connexion between computer graphics and the movie industry, and for good reasons: computer generated images have gradually become a very comfortable alternative to hand-made animatronics and movie sets, crowds, and even individual actors. Not only does a digital alternative allow to endlessly explore the possibilities of easy modification of the sets, shooting new view angles and lighting, but it also allows to simulate otherwise costly processes such as the destruction of the model. Interestingly, geometric models of creatures involved in movie making are often generated through a tedious digitalization process of real models made by artists out of clay.

The movie "Tron" (1982) was the first notable use of computer graphics in a movie. Jurassic Park (1993) showed photorealistic fully animated computer graphics creatures for the first time. Before that, creatures in movies mostly relied on animated puppets (a.k.a. animatronics) such as in Alien [1] The Grail is of course to

[1] The notion of "photorealistic creature" is itself absurd since all these creatures cannot actually be compared to anything real. I believe that in Jurassic Park, dinosaurs were mostly visible under the rain, simply because realistic wet lizards required simpler shading models than dry ones.

Figure 1.1: *Typical special effects where the actual shot is performed in front of a green screen. In post production, numerical simulation and animation are used to generate the snow that is seamlessly blended into the movie. Images via SpinVFX.* `http://www.spinvfx.com/work/game-of-thrones/`.

represent human characters in a way that is seamless for the spectator. This is extremely challenging because, as human beings, we're ultimately trained to recognize our own species. Many attempts could be cited. The Curious Case of Benjamin Button (2008) is a one example of such a successful achievement in this particular direction, not only because it features very technical facial animation work, but also because it does not stand in the line of science fiction movies where you would most naturally expect such technology to be involved. Nowadays some movies are almost entirely done with computer generated images. Avatar (2009) is a one of them.

Figure 1.1 shows a typical example of integration techniques in the process of using a green screen during the actual shot, and adding a computer generated set afterwards. Seamlessly integrating computer-generated actors/objects and sets into a movie requires modeling and acquisition of the data, simulation of physical phenomena, and image combination techniques, all of which should deal with huge amounts of data, and therefore require an equally large computational power. The role of research in this field is to provide efficient algorithms to perform these tasks while handling the complexity.

Because of the race for realism and visual complexity, research in computer graphics works hand-in-hand with movie production companies in order to design new algorithms and data models. Some of the most successful actors of the movie industry (Pixar, Digital Domain, etc) have their own research departments in the field of computer graphics, and they develop their own software suites in order to master the complete production pipeline (An interesting reference is [Christensen 2012]). Even algorithms developed by academic researchers have a real opportunity to take place in today's movie production, if they are easy enough to adapt to a production level. This is probably one reason for the implicit "production ready" requirement of some conferences like Siggraph.

Computational photography refers to the set of automatic treatments of a digital photograph or an entire set of pictures. The aim is to produce a result that is not necessarily possible to obtain from an ordinary camera. Examples include:

panorama stitching, range compression (a.k.a. tone mapping), filtering for image enhancement or artistic purpose, color and histogram transfers, focus or defocus operations, motion blur removal, image compression, etc. Most of these techniques come from the world of image analysis and processing, but can also be considered to stand in the field of computer graphics. They all involve very advanced image manipulation and filtering techniques, that are the result of years of research. Recently a significant amount of rather technical work has shown to serve very practical applications. A typical example is the action to "remove" an object within an image while automatically filling the place in a consistent way [Hays 2007]. Image resizing algorithms [Avidan 2007], image re-targeting [Farbman 2009, Chen 2013], intrinsic image decomposition [Bousseau 2009, Carroll 2011], and edge-aware image smoothing [Karacan 2013a] have shown truly impressive results as well.

Virtual prototyping is another important application to computer graphics that is highly demanding of efficient algorithms. The purpose of virtual prototyping is to validate a product design using computer simulation and visualization. It involves the design and acquisition of geometry, material and illumination models, the simulation of undergoing physical processes, and the display of the results. Between 2002 and 2006, I participated in the RealReflect European project [2], which goal was to build a full virtual prototyping pipeline for cars. The project was a collaboration between eleven academic and industrial actors (including Daimler in Germany, and INRIA in France). RealReflect was a representative example of the use of computer graphics techniques for virtual prototyping, since computer graphics research teams were involved in all possible stages of the pipeline: acquisition and representation of light sources, car geometry and material properties; lighting simulation inside the cars; and rendering in a virtual reality cave using a proper tone reproduction system. All these techniques have been the subject of a lot of research in particular to efficiently cope with the huge amount of data involved.

One important problem brought on the table by virtual prototyping is that the topological and geometrical quality of mesh models need to be consistent with the algorithms that are used for physical simulation. Finite element methods for instance are much more stable on meshes with nearly equilateral triangles, which require careful remeshing techniques [Botsch 2010].

Data manipulation and visualization. Most of the industry works with computers, and requires to visualize all sorts of data. Research problems in computer graphics that are relevant to data visualization are numerous: the efficient display of gigantic geometric scenes requires clever occlusion culling algorithms to avoid unnecessary handling of geometric primitives [Correa 2007]. Automatic placement of geometric primitives (such as when displaying a large graph [Munzner 2000]), automatic labeling [Been 2008], and generally the selection of what is displayed are non trivial problems as well.

[2] `http://cg.cs.uni-bonn.de/en/projects/realreflect/`

Properly accounting for the display technology is sometimes a real challenge since displays usually have a much lower dynamic range than the scenes they are supposed to show [Yoshida 2006]. A typical field of application is simulators where the effect of the sun and reflexions on the windshield of a car for instance need to be rendered as realistically as possible.

In some situations it might even be necessary to generate additional data on-the-fly in a way that is consistent with the real data, in order to fill in missing information. This can be achieved using procedural content generation methods [Müller 2006, Bruneton 2012].

Video games have a major connexion with research in computer graphics. First of all, just like movies, video games need the production of computer generated images. The essential difference is that whereas the movie industry can afford to spend hours and an enormous computational power to generate photorealistic images, video games need them fast (By that I mean really fast). The usual outcome is to trade realism for speed, while trying to keep the best possible ratio between the two. That is why research in computer graphics is useful in the domain of video games. One example is the computation of approximate global illumination in real time [Kaplanyan 2010], because fast enough physically based global illumination remains out of reach.

The production of digital assets for video games is also a challenge. The usual size of data for a video game is most of the time several gigabytes, among which the game logic and display algorithms are a tiny proportion. Most of this data is actually geometric models, textures, level maps, etc. Generating them takes a lot of effort to artists, and it is therefore a field of research to produce algorithms to help generating them automatically. Texture synthesis by example [Efros 1999] and procedural generation of vegetation and terrain [Génevaux 2013] are such examples. Handling very large collections of geometric models is also a challenge, at which various techniques can prove very useful, for instance practical selection algorithms [Eitz 2012].

Serious games are now becoming more popular. A quick look at the spectrum of games which have an educational purpose shows an incredible number of application domains such as defense, education, scientific exploration, health care, emergency management, city planning, and engineering. However, because serious games are not primarily intended to provide entertainment, their demand is oriented toward accurate simulation of physical phenomena.

Art and communication. General conferences such as Siggraph have seen lots of contributions related to art, which inherently solve a "toy problem" with an application to the generation of artistic results. Examples include pen-and-ink [Wilson 2004] and painterly [Hertzmann 2000] illustration, procedural generation of patterns [Wong 1998], generation of shadows [Mitra 2009], etc. Although the action of re-creating by a computer program the style of an existing artist may not

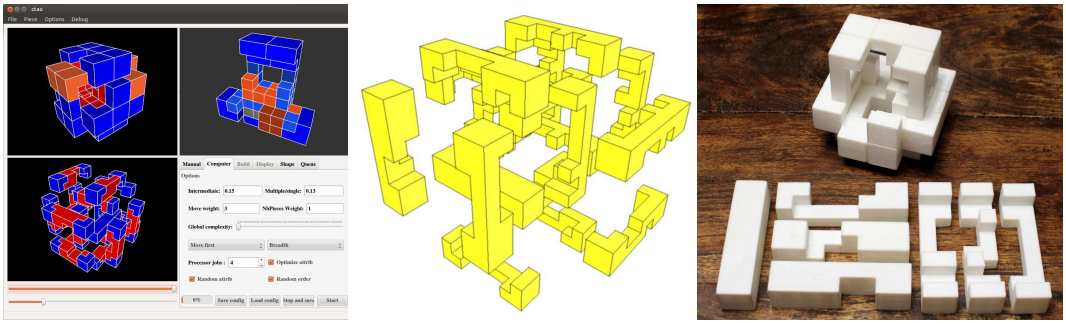


Figure 1.2: *3D Printing example. The program on the left was used to automatically generate a 18-pieces burr puzzle, which pieces are shown in the middle. A 3D printer was afterwards used in order to get the real object in hands, at right.*

seem always useful, it can be an interesting opportunity to raise more fundamental questions, some of which are still unsolved. One good example is style capture: is it possible to automatically recognize which painter produced a given painting? Another example is to find compromises in the unsolvable problem of temporal coherence that arises when drawing 3D objects on a 2D canvas [Bénard 2011]. Finally, computers are a new space of creativity, which gives lots of room for real artists to create. Because these artists will probably use computer graphics techniques that have been previously developed by researchers, one can say that research in computer graphics also serves graphical arts to some extent [StãNculescu 2013].

3D printing is a newly emerging application to computer graphics. It allows rapid prototyping with side applications in artistic design and health care. Anything can actually be printed including jewelry, food, consumer replacement parts, and even buildings. It is currently considered to be the upcoming revolutionary technology for the 21th century. Figure 1.2 shows such an example where a 3D burr puzzle was automatically generated by a computer program before being printed for real. Various research problems are related to 3D printing, most of which deal with geometric optimization. Here's some examples: (1) balance problem: one wants to make printed models stand, by slightly deforming the object [Prévost 2013, Hergel 2014]; (2) robustness: computing weaknesses in printed shapes [Zhou 2013] and adding internal structures so as to make the printed object more robust, while limiting the total amount of matter that is used.Typical solutions involve the generation of bee hive structure inside void areas; (3) solving geometric constraints of the printing technology: printing isn't always possible due to the angles and details of the input model. Therefore it is necessary to split the model following intrinsic constraints of the printing material, which can be a geometric challenge. Various extensions of 3D printing are currently under investigation such as BRDF printing [Matusik 2009]. Also related to 3D printing is the problem of watermarking of 3D digital models [Wang 2007b].

Virtual and augmented reality are probably about to explode in everyday life. This is a consequence of the successful miniaturization of graphics hardware which are currently on the verge of bringing the accessibility of these techniques to everyone. The Google glass are a perfect example of such a technique. The constraints that come with miniaturization require algorithms to be even more efficient. Similarly, Valve (a company that creates video games) is currently exploring the possibility of using virtual reality devices such as the Occulus Rift [3] for its games. The connexion between research in computer graphics and virtual reality is tight, and partly driven by the need for interactivity toward the changes in display conditions that are quite unpredictable, and maintaining quality in the displayed images [Pohl 2013].

In summary, all the application domains I just reviewed are an incredible source of problems to be solved by research in computer graphics, some of which are really challenging.

1.2 My perception of research in computer graphics

Research in computer graphics ties together a rather large spectrum of sub-domains that share a common goal: facilitate the generation of synthetic images. These sub-domains correspond to the various steps involved in the process of image creation: generation and processing of geometry and photometric data; Modeling; Real time rendering and offline global illumination; image treatment and computational imagery. Some sub-domains are emerging, such as computational photography; some are already quite mature, as is it the case for lighting simulation and geometry processing.

Computer graphics is a particularly active domain for three reasons: (1) first, the constant evolution of technologies involved for display, computation, printing, and acquisition of data raise new questions and research topics on a yearly basis. It happens regularly that a new technology brings space for improving existing computational methods that had previously been left aside because their inherent cost could not previously be handled with current hardware. Algorithms must sometimes be reshaped in order to fit the programmability constraints of new hardware, as it has recently been the case for programmable GPUs; (2) Secondly, one side effect of the constant increasing of computational power is the increasing demand in data complexity. Therefore, whatever the hardware to solve existing problems, the data that comes with them always raises the cost to the acceptable limit, meaning that researchers must always fight to improve the computation cost of existing methods; (3) Finally, computer graphics stands at the convergence of multiple domains: physics, geometry, algorithmics, mathematics, perception and vision, which makes the research inherently cross-domains.

The goal of a researcher in computer graphics is generally to produce algorithms

[3] See this: `http://www.academia.edu/5387318/The_Oculus_Rift_and_Immersion_through_Fear`

to compute images visible by human beings in an interactive application. That certainly ties the research to a specific set of working principles: computation must be as fast as possible, which means being able to perform controlled approximations. Algorithms must be scalable, which mostly means working with hierarchical methods. The result itself must be tuned to fit the characteristics of the human visual system: our eyes are more sensitive to some particular types of errors which are not easily characterized by the simplest mathematical distances. Finally, one generally needs to aim for simple and robust solutions, and allow as much precomputation as possible. All the research that I am presenting in this document is driven by these requirements.

Connexion to other scientific domains

Mathematics are essentially used in computer graphics as a tool [Vince 2010], the principal aspects of which are linear algebra, probability and statistics, differential and integral geometry and numerical computation methods. Some sub-domains such as computational geometry should certainly be counted in the list of contributions of research in computer graphics since computer graphics need the handling and processing of geometric models with very specific constraints. A number of work pieces in computer graphics are therefore really contributing to the mathematics of geometry processing and differential geometry, that are required by mesh filtering, interpolation, reconstruction, etc. Similarly, advances in understanding the properties of the light transport operator might be seens as contributions to the knowledge of the Fredholm operator for a particular class of discontinuous kernels.

Applications to computer graphics often need to simulate physical phenomena, even in a simplified way. Therefore, one way to proceed is to start from the physical models and perform controlled approximations. As a consequence, computer graphics commonly use notions of mechanics, optics, electromagnetics, and thermal exchange, and contribute to consistently simplifying them.

Computer vision has strong connections with computer graphics [Lengyel 1998]. Computer vision for instance develops techniques for understanding and acquiring information from measurements by a captor. Some of these techniques are strongly related to the problems that material and geometry acquisition techniques in computer graphics are dealing with.

Perception is also a connected research domain [Thompson 2011]. Computer generated images often need a compromise between computation cost and accuracy, in order to be generated within a reasonable amount of time. Because these images are ultimately viewed by the human visual system, it is important to tune the measures for accuracy in accordance to our own visual system.

Current trends

Computational power increases but the demand for scene complexity increases as well. It seems that the demand in complexity and realism always goes a little bit

beyond the current storage and computational capability, which keeps pushing the research to develop more efficient algorithms. To understand this literally, it is important to realize that the growth in computational power of graphics processors has almost every time exceeded Moore's law. When I was at ENSIMAG in 1996, my teacher in numerical analysis, Francois Robert, used to talk about the "latest Cray machine" that had the incredible power of 1 Teraflop. That same Teraflop is currently the computational power of the graphics card of my laptop. It is also common today to distribute computation on farms of computers each having multiple processors. As suggested by the citation of Jim Blinn in the beginning of this chapter, the race between computational power and scene complexity is still going on.

Real time simulation of complex physical phenomena for display purposes is a major problem. While the problem of physically accurate global illumination for instance can mostly be understood to be already solved, real-time global illumination is still out of reach. Because scene complexity keeps increasing, it is likely that at any time we will never reach real time global illumination for the typical scenes that are used at that time. For a given level of scene complexity however, real time global illumination will be reached eventually. The same applies to large scale fluid simulation, natural phenomena, etc. As a consequence, the quest of researchers in computer graphics is mostly about scalability of algorithms [Forrest 2003].

Large scale data acquisition is another side effect of the increase of the required complexity. With the ever increasing demand in realism (especially from the movie industry) comes the need for larger and more accurate digital models in general. These models can be of geometric or photometric nature. Acquiring these models usually involves complex treatments such as filtering, 3D registration, compression, etc.

Human interaction with computer generated media is also an important aspect that will need lots of research in the future. Virtual and augmented reality are a source of many unsolved problems in this field, for instance how to optimize computation according to unpredictable movements of the user.

So, why do I like it?

First of all, research in computer graphics produces nice images [4]. Beyond the satisfaction of producing a result that might be visually pleasing, the ability to "see" an algorithm in action is very rewarding.

Variety is certainly an important aspect as well: because computer graphics is related to so many different research subjects, it is easy to switch between geometry processing and image analysis which are totally different fields, while using similar tools in different contexts (e.g. wavelets, Fourier analysis).

Researchers in computer graphics have the opportunity to manipulate very interesting mathematical tools. Among my favorites are spherical harmonics and mesh-

[4] Even when a program does not work, the faulty result is sometimes surprisingly interesting. See for instance `http://www-evasion.imag.fr/~Fabrice.Neyret/perso/bugs.html`

based wavelets. I also enjoy implementing and working with numerical methods, especially for integration. Monte-Carlo algorithms that are used in global illumination also have some kind of magic power, since you add up billions of random values in order to compute a result that is eventually not random at all. Implementation

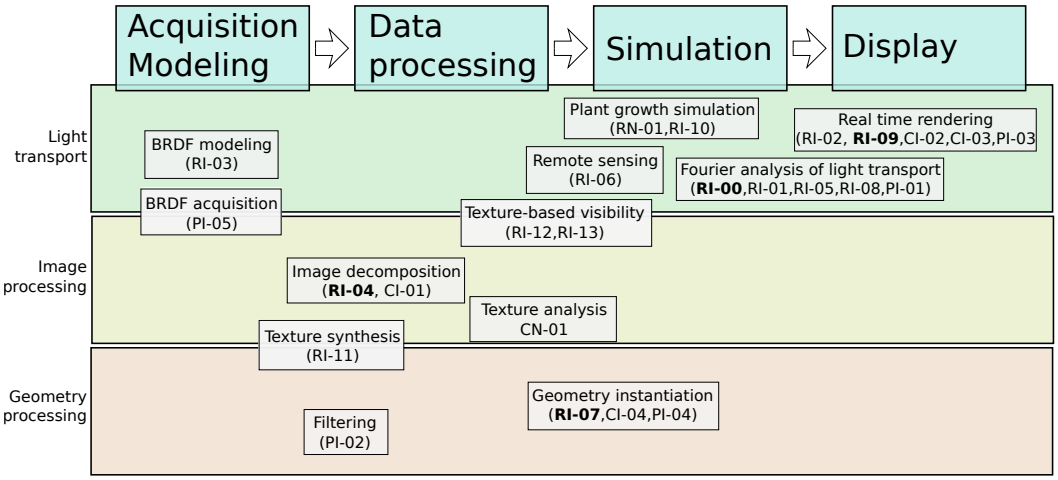


Figure 1.3: *Computer graphics pipeline, with the different sub domains I contributed to, sorted with respect to light transport, image and geometry processing. All references are listed in Appendix A. Reference in boldface are appended in Appendix B. All references are available in PDF format at* `http://maverick.inria.fr/Publications/index?author=soler`.

aspects are really compelling, especially for hierarchical algorithms, parallel and SIMD programming, although programming on GPU is sometimes particularly tedious and difficult to debug. Beyond that, we're often working with very recent graphics hardware that is constantly evolving. Some research in computer graphics have even influenced the future of existing graphics hardware [Eisemann 2007].

Finally, as shown earlier, there is a enormous set of application domains, and therefore the opportunities to collaborate with the industry are really significant and compelling.

1.3 Position of the presented work

I contributed in different sub-domains of computer graphics: simulation of light transport, geometry processing and image processing. In each of them, my contributions stand at various places in the pipeline of generating synthetic images: data acquisition and representation, data processing, physical simulation and display. Figure 1.3 summarizes these connections and displays the various fields in which I have contributed.

In Chapter 2, I list my academic activities, while in Chapter 3, I will summarize the research I have been involved in since my PhD thesis. Appendix A lists

all my publications. In appendix B, I appended a representative subset of these publications.

Chapter 2

Scientific activities

I list in this chapter the students I have advised, the projects I have been involved into, and additional scientific activities such as paper reviewing and code dissemination.

2.1 Past PhD students

For each student I shortly describe the research project during the thesis, and give the percentage at which I was advising the student with a co-supervisor.

Benoit Zupancic (2012–2015) *Reflectance Acquisition using Compressive Sensing* (95%, with N. Holzschuch).
The goal of this thesis is to explore the possibility to leverage the intrinsic sparsity of reflectance functions to design acquisition methods based on the recent theory of compressive sensing **[PI-05]**.

Laurent Belcour (2009–2012) *Frequency Analysis of Light transport* (95%, with N. Holzschuch).
Currently on a post-doc at University of Montreal.
The goal of the thesis was to extend the work done by Durand *et al* **[RI-08]** to the time domain so as to handle motion blur. We eventually found a very interesting representation for the spectrum of local light fields based on the covariance matrix of the power spectrum, which opened very interesting avenues and produced fruitful results and significant publications **[RI-01,RI-02,CI-02,PI-01]**.

Mahdi Bagher (2009–2012) *Real-time rendering of complex materials* (60%, with N. Holzschuch).
Currently employed by Microsoft in Vancouver.
The thesis was dedicated to efficiently rendering measured materials, which we achieved through two very different contributions. First we simplified the theory of the Fourier analysis of light transport so as to use it directly on the GPU **[RI-02,CI-02]**. Then we derived a new analytical approximation of materials based on a shifted gamma distribution that proved to overcome existing approximations while offering good computational properties **[RI-03]**.

Pierre-Edouard Landes (2007–2011) *Analysis and synthesis of textures with stochastic distributions of shapes* (95%, with F. Sillion).

Currently engineer at Aerys, a french SME.
We studied the problem of automatically recognizing distributions of shapes in an image, with potential overlaps, so as to later perform statistics on the distribution of these shapes to be able to generate a similar texture **[CN-01]**.

Aurélien Martinet (2003–2007) *Automatic instantiation of geometry* (90%, with F. Sillion).
Currently an engineer at Tetrane.
We studied the problem of automatically instancing incoherent geometry that is available as a list of simple geometric primitives such as triangles. Part of the work was to find an automatic method to deterministically determine symmetries of shapes, which we further applied to design a constructive algorithm to find similar objects in a scene **[PI-04,RI-07]**.

2.2 Undergraduate students and engineers

Isabelle Delore (2009–2011) *Engineer, GARDEN Project.*
Methods for automatic content generation in video games.
The GARDEN project, in collaboration with EDEN Games—a French video game company—was aiming at developing a full pipeline for the design of video games. Isabelle Delore worked on evaluating existing algorithms including procedural noise from example, image compression, etc.

Olivier Hoel (2007–2009) *Engineer, GENAC II Project.*
GPU implementation of illumination methods for video games.
I advise O.Hoel in the context of the GENAC II project which aimed at developing efficient algorithms for content generation in video games. Olivier worked in particular on the problem of computing an approximation of indirect illumination in screen-space, in a deferred shading pipeline (See Section 3.3.2).

Pierre-Edouard Landes (2005–2006) *DEA.*
Analysis and synthesis of textures with stochastic pattern distributions.
P-E.Landes did his DEA on that subject which he pursueded during his PhD. The goal was to automatically extract distributions of sprites from an image, using pixel-based similarity measures and graph completion.

Jean-Christophe Roche (2004–2006) *Engineer, RealReflect Project*
Implementation of a complete lighting simulation platform. I advised this person as an engineer to develop a complete global illumination system, including geometry management based on an extension of X3D, material representations (BRDFs and BTFs), light sources including the full IESNA standard, global illumination using photon mapping, scene management and distributed computation tools. Later this code has been entirely rewritten and is used by my students (See Section 2.7).

Stéphane Guy (2004–2005) *DEA*

Real-time rendering of birefringent materials

During this project we explored the possibilities of rendering facetted gemstones in real-time on current graphics hardware. For that we derived controlled approximations of the full optics of birefringent materials and simplified them to become tractable on GPU (See Section 3.3.1). Afterwards, Stéphane Guy did a PhD on a totally different subject.

David Roger (2003–2004) *DEA*

Analysis of diffuse intereflexion on the ambient light intensity

The goal of this work was to figure out in what ways diffuse intereflexion in a scene contributes to the ambient intensity in this scene, and propose heuristics to compute visually pleasant yet non physically based ambient illumination knowing the materials that compose a given scene.

Thomas Schneider (2003–2004) *DEA IVR*

Expressive rendering of global illumination

In the beginning of years 2000, a lot of papers have been published on the subject of expressive rendering (a.k.a. non photorealistic rendering). None of them however explored the possible visual clues to properly render global illumination. that was precisely the goal of this research.

Benjamin Sergeant (2002–2003) *DEA IVR*

Extraction of semantic information in a 3D scene using Bayesian classification.

This work was a preliminary investigation to our later work on instantiation. The idea was to try to recover object classes from the topology of the graph that connects the different parts of an object, using Bayesian methods.

2.3 Funded projects

2011-2015 ALTA, *ANR Blanc.* `http://maverick.inria.fr/Projects/ALTA/`.

The ALTA project stands for Analysis of Light Transport operator and Applications. It is funded by the french National Research Agency (ANR), and ties together three teams of INRIA: MAVERICK (INRIA Rhône-Alpes), REVES (INRIA Sophia) and MANAO (INRIA Bordeau). These teams collaborate in the triple goal of performing a frequency, dimensional and first order analysis of the light transport operator. The targeted applications are both real time rendering and global illumination. The project is managed by N. Holzschuch, and I'm responsible for conducting the research on frequency and dimensional analysis of light transport.

2009-2011 Garden, *FUI*, Région Rhône-Alpes et Pôle de Compétitivité Imaginove.

This project teamed up academic actors including INRIA and the LIG laboratory, and video game companies such as Eden Games, Krysalide and Gamr7. The goal was to develop a full game development pipeline. My contribution was to manage the part related to illumination, advising sudents and engineers (Isabelle Delore, Olivier Hoel, Pierre-Edouard Landes) for designing new techniques and implementing them on graphics hardware.

2007-2009 Genac II, *FUI, Région Rhône-Alpes et Pôle de Compétitivité Imaginove.*

The GENAC II project deals with automated content generation for video games. This applies to generating motion for non playing game characters, generating textures, game levels, vegetation, and performing light simulation on top of this. I was mainly involved in the "illumination" side of the project, advising an engineer (O. Hoel) for 2 years.

2001-2006 RealReflect, *European project IST-2001-34744.*

RealReflect was a long-term European research project. Academical partners included INRIA, University of Bohn, University of Wien, and university of Prague. Industrial partners were Daimler Chrysler (Germany), and Faurecia (France). The goal was to design a complete pipeline for virtual prototyping in the car industry. This included acquisition and treatment of material properties, illumination and geometry parts involved in the creation process of a car. Then lighting simulation was performed in order to realistically render the prototypes. The display was also tuned to fit the requirements of a cave system offering a 3D head-tracking display. I was involved in leading work package 7 (lighting simulation) in collaboration with F.Sillion, and I was advising an engineer to develop a lighting simulation platform.

1999-2000 Soleil, *Action de Recherche Coopérative INRIA.*

This project was a collaboration between LIAMA (French-Chinese Laboratory for Applied Mathematics and Automatics, Beijing, China) and INRIA. Its goal was to design a physiological plant growth simulator by connecting a plant growth simulator with a global illumination engine, the former computing the geometry of the plant at the next growing step given the distribution of light inside the plant that is calculated by the later. This project gave birth to plant models which shape and distribution of leaves are influenced by light (See examples on the project page `http://maverick.inria.fr/Projects/Soleil/index.html`). My involvement was to handle the simulation of radiative exchanges in the plants.

2.4 Other collaborations

2013-2014 University of Montreal, with Derek Nowrouzezahrai
In 2010, I had the idea of using rotated zonal harmonics as a new basis for representing harmonic functions on the sphere. Derek had the same idea and published it before me. So I joined him to work on that subject. We're currently in an active collaboration on the subject of spherical filtering.

2010-2014 Cornell University, Ithaca, with Prof. Kavita Bala.
Laurent Belcour, a former PhD student, has spent a few months during his PhD at Cornell University, which was the opportunity to work with Prof. Kavita Bala on Frequency analysis of participating media **[RI-00]**.

2011-2012 University College London, UK with Prof. Jan Kautz and Kartic Subr.
Kartic Subr was a postdoctoral researcher at ARTIS for one year, then he left for UC London to work with Jan Kautz. Having a lot of common research interests we collaborated on the project of accurate binary image selection using inaccurate input **[CI-01]**. This work was also a collaboration with Sylvain Paris, from Adobe Research.

2005-2010 MIT, Boston, with Prof. Frédo Durand.
Fredo Durand is a former PhD student of C.Puech who defended his PhD at Grenoble University. He is now a professor at MIT. We have been collaborating on various subjects including frequency analysis of light transport (of which he also owns the original concept), and image analysis **[RI-01,RI-04,RI-05,RI-08]**.

2002-2004 Noveltis, Toulouse.
Noveltis is a SME which activities are focussed toward bridging the gap between the industry and researchers. They develop solutions for the former using the knowledge of the later. I have worked with Noveltis in collaboration with V.Bruniquel on radiative transfer simulation in vegetation for remote sensing**[RI-06]**.

1999-2000 LIAMA, Beijing, China. with P.Dereffye
P.Dereffye is a international-class researcher in the domain of plant architecture, currently invited professor at the CEF (*Centre d'Etude de la Forêt*). We have worked together on lighting simulation for physiological plant growth simulation, in the context of the *SOLEIL* project.

2.5 Program committee and reviews

I've performed several reviews for the following journals and conferences: Siggraph (since 1998), Eurographics Symposium on rendering (since 1998), Eurographics Con-

ference, IEEE Transactions on Visualization and Computer Graphics, ACM Transactions on Computer Graphics, and the Visual Computer.

I have participated in the following program committee: Eurographics (2004, 2013), Eurographics Symposium on Rendering 2014, Pacific Graphics (2012, 2013).

I have been solicited a few times for ANR projects reviewing.

In 2014 I co-chair the tutorials for the Eurographics 2015 conference with Matthias Zwicker.

2.6 Consulting at Digisens S.A.

Digisens is a SME currently employing 6-7 persons which sells solutions for efficient computer tomography. They provide a software suite to perform many of the operations along the acquisition pipeline: tomographic reconstruction on the GPU, data filtering, mesh production, metrology, etc.

I've acted as a consultant for Digisens SA[1] for 5 years (from 2007 to early 2013). My work consisted into solving scientific problems of very various nature, performing technology intelligence and providing proof-of-concept code. I've been working successively on mesh extraction, automated volume stitching, mesh filtering, mesh and volume segmentation, volume filtering and removal of tomography artifacts, and compressive sensing tomography.

The work I conducted at INRIA about filtering 3D tomographic volumes was partly inspired by my involvement at Digisens **[PI-02]**.

2.7 Code

Along all these past years, I have accumulated a significant amount of code, some of which has been kept in a portable form. I'm only listing three representative code projects here:

HQR The High Quality Renderer is a rendering platform that was first created during the RealReflect project. It has been extended with a plugin system that allows to change every internal component, allowing user-customed materials, sampling algorithms, illumination sources, and geometry representation and management methods, to be used in place of the existing. It is a very practical tool for experimenting with new light simulation algorithms while inheriting the basic components to process geometric scenes, image output, etc. See `http://maverick.imag.fr/Membres/Cyril.Soler/HQR`.

VRender The VRender library is a simple tool to render the content of an OpenGL window to a vectorial device such as Postscript, XFig, and SVG. The main usage of such a library is to make clean vectorial drawings for publications, books, etc. In practice, VRender replaces the z-buffer based hidden surface

[1] `http://www.digisens3d.com`

removal of OpenGL by sorting the geometric primitives so that they can be rendered in a back-to-front order, possibly cutting them into pieces to solve cycles. VRender is also responsible for the vectorial snapshot feature of the QGLViewer library (See `http://maverick.inria.fr/Software/VRender/`).

Spherical harmonic library This library allows to perform all sorts of useful tasks with spherical harmonics. It implements my own—currently unpublished—SH rotation formula, as well as the famous ZXZXZ method [Kautz 2002]. It also contains the necessary tools for stable computation of SH at high orders (L>150) This library has been improved over the years with a significant amount of test code, IO methods and visualization tools. This code is mostly used by students in the Maverick team. Although this code is currently not public, I am planning to release it so that other researchers can use it.

Chapter 3

Research summary

In this chapter, I summarize the research I did after my PhD thesis in 1998. Although my preferred subject in the early years has been the simulation of radiative exchanges, I gradually moved to other areas of the domain of computer graphics: texture synthesis, geometry and image analysis. While exploring these fields, I continued to work on the problem of global illumination, with the Fourier analysis of light transport and its following contributions. At the present, I'm heading toward material acquisition and computational photography. Figure 3.1 below summarizes this by listing the papers I had in international conferences and journals, along with the connexion of the various co-authors I collaborated with.

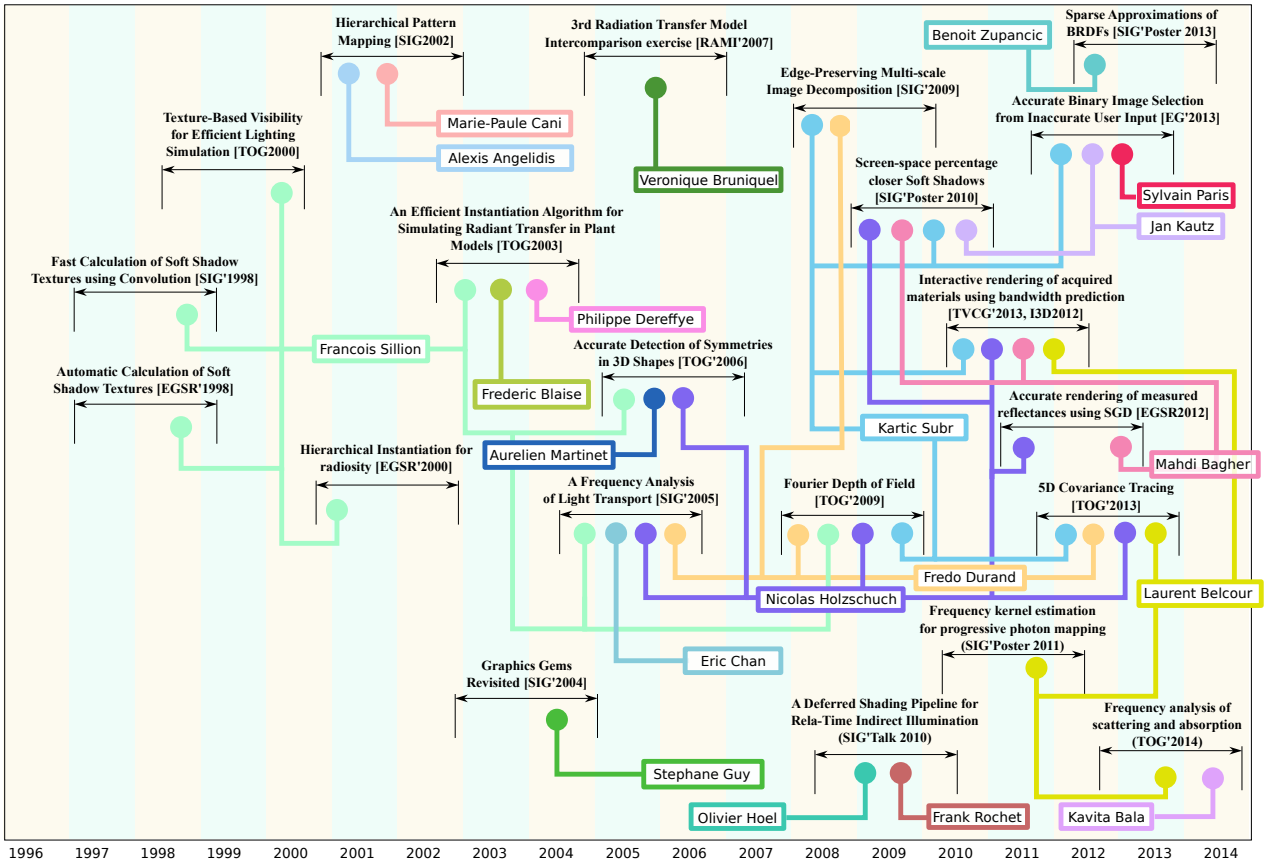


Figure 3.1: Co-authors and timeline for my publications into international journals and conferences

A full list of my publications is available in appendix A. They are referenced throughout the summary of research that I present below.

3.1 Simulation of radiative exchanges

The term *global illumination* designates the problem of computing the distribution of light in a geometric scene, accounting for how it is emitted from light sources and how it interacts with materials and objects. When the goal is to make synthetic images, one needs to complete such a simulation by a measurement of how light energy is perceived by a virtual camera. For some applications however, it is the distribution of radiant energy in the scene that constitutes the ultimate goal of the calculation. In any case, because the radiant energy that is received at each point in a scene partly reflects and influences the radiant energy that is received at all other points, the problem of computing the distribution of light in a scene essentially is to solve a global equilibrium equation.

I've been working on the simulation of radiative exchange for almost fourteen years now, on top of approximately ten years of prior work. It is not possible to detail the huge amount of existing contributions in this field. Each of my papers in this domain detail the relevant subset of previous work. In this section however, I'm giving the reader the necessary bits to understand the motivation of my contributions. I'm starting with a brief introduction to the physical aspects of global illumination and radiative exchange simulation in general, explaining what important questions needed further work, and I give an overview of how I contributed to research in this domain.

3.1.1 The rendering equation

For all applications considered in this document, light is supposed to reach equilibrium instantly. Light is represented by a quantity $L(\mathbf{x}, \omega)$ called *radiance*, that is the amount of energy propagating at point $\mathbf{x}$ in direction ω, per unit of solid angle and area orthogonal to ω. Light generally interacts with surfaces and volumes where it reflects and diffuses. The interaction with surfaces can be modelled by the Bidirectional Reflectance Distribution Function ρ of the material, also known as BRDF, which ties the distribution of incident radiance L_i at $\mathbf{x}$ to the reflected differential radiance in direction ω:

$$dL(\mathbf{x}, \omega) = \rho(\mathbf{x}, \omega, \omega') L_i(\mathbf{x}, \omega') cos\theta d\omega'$$

In this equation, θ is the angle between the surface normal and ω'. Given these quantities, it is easy to express the equilibrium of light energy in a scene, by writing that the energy at each point x is the sum of the energy emitted by that point, plus the integral of radiance coming from all other points, that reflects at $\mathbf{x}$:

$$L(\mathbf{x}, \omega) = L_e(\mathbf{x}, \omega) + \int_\Omega \rho(\mathbf{x}, \omega, \omega') L_i(\mathbf{x}, \omega') cos\theta d\omega' \tag{3.1}$$

Since L_i and L are the same quantity expressed using two different parameterizations, this equation is intrinsically a linear integral equation (Fredholm equation of the second kind), which in general does not admit an analytical solution [Kajiya 1986].

Because the integral in the second member is a linear transform of the distribution of radiance L, it is common to define it as a linear operator over the infinite dimensional space of radiance distributions: the *light transport operator* $\mathcal{T}$. With this compact notation, the radiance equation becomes:

$$L = L_e + \mathcal{T}L$$

This formulation emphases even further the equilibrium nature of the radiance equation. Physical considerations show that the spectral radius of $\mathcal{T}$ is strictly smaller than 1. Indeed, if $\mathcal{T}$ has an eigenvalue of modulus greater than 1, it would be possible to create a scene where the total light energy strictly increases at each bounce, breaking the law of energy conservation. If $\mathcal{T}$ had an eigenvalue of modulus equal to 1, we should be able to create an object inside which a non zero distribution of light bounces indefinitely [1]. Consequently, L can be expressed using the following sum, that is converging:

$$L = (\mathcal{I} - \mathcal{T})^{-1} L_e = \left(\sum_{k=0}^{\infty} \mathcal{T}^k \right) L_e \tag{3.2}$$

It is notable however that very little work has been done in studying the spectrum of $\mathcal{T}$ and its eigen functions in the context of computer graphics for the full radiance equation [Baranoski 1997, Ashdown 2001, Lessig 2010], meaning with the particular kernel based on general reflectance functions. Now I will briefly present two families of methods for solving the radiance equation: Galerkin methods and Monte-Carlo methods, since my contributions are built on top of these.

3.1.1.1 Galerkin methods

Galerkin methods have been the primary option for solving Equation 3.1, better known as the *radiosity methods* [Sillion 1994]. In these, the light distribution L is approximated by an element of a subspace of functions of finite dimension, and $\mathcal{T}$ is approximated by a matrix T that operates onto coefficient vectors of a basis in this subspace. This is an approximation indeed, since there is no reason for the space spanned by the chosen basis to be stable by $\mathcal{T}$.

The simplest approach is to choose a basis of functions that are piecewise constant, that we define with respect to a mesh of the space where L is defined. For the most general case, a mesh element will be a 4D sub-region of surfaces and directions. In the classical *radiosity* method, L is supposed to only depend on the position in the scene, which corresponds to having all materials be perfectly diffuse. In this case L is represented by its vector $\mathbf{l}$ with coefficients $\{l_i\}_{i=1..N}$ such that:

$$T\mathbf{l} = \mathbf{l}_e + T\mathbf{l} \quad \text{with} \quad T_{ij} = \frac{\rho_i}{A_i} \int_{A_i \times A_j} v(\mathbf{x}, \mathbf{y}) \frac{\cos\theta \cos\theta'}{\|\mathbf{x} - \mathbf{y}\|^2} d\mathbf{x} d\mathbf{y} \tag{3.3}$$

[1] One can imagine a sphere internally covered with a mirror, but injecting light inside this object will necessarily create a hole, or an obstacle, hence reducing the eigenvalue

In this equation, ρ_i and A_i are the diffuse reflectance and area of surface element i. The angles θ and θ' are the angles between the surface normals and the line joining $\mathbf{x}$ and $\mathbf{y}$, and $v(\mathbf{x}, \mathbf{y})$ is the binary visibility between these points. Because v can be arbitrarily complex, there is no analytical formula to compute the form factors T_{ij}, and one generally needs to rely on numerical integration. For this reason, computing the form factors is by far the most computationally expensive task in solving Equation 3.3.

Although it is very limited in the scope of image synthesis because of its poor ability to represent non diffuse materials, the radiosity method has definite applications in the field of heat transfer calculations. The main issue is the quadratic cost of computing the form factors. In other words, if a N-elements discretization of the scene does not bring enough accuracy, half-splitting each element will raise the number of elements to $2N$ and raise computation cost by four times.

To overcome this particular issue, researchers have proposed a hierarchical formulation of the radiosity equation, where form factors and light exchanges are computed between arbitrarily large groups of surfaces (and objects), which size is adapted so as to keep the error below a user-defined threshold [Smits 1994, Seidel 1997]. This brings the simulation cost down to $\mathcal{O}(N \log N)$.

Despite this definite improvement, the memory cost of the radiosity method stays proportional to the accuracy at which the solution is computed, which makes it totally intractable for even moderately large scenes. One solution would be to use instantiation, which means replicating self-similar geometry using references and geometric transforms. This is particularly suitable to architectural scenes where lots of furniture items are replicated, and vegetation scenes because plants naturally display self similarity [Prusinkiewicz 1990]. However, the different instances of a given piece of geometry are likely to receive a different amount of light, which means they will need a specific storage and a careful handling of this energy at an appropriate hierarchical level in each instance.

For this reason, I worked on the problem of automatic instantiation of geometry (Section 3.4.2) and the use of instantiation in the hierarchical radiosity method (Section 3.1.2). Being successful in this field, we applied these algorithms to the simulation of light exchanges in vegetation scenes, for the application to plant growth simulation and remote sensing.

3.1.1.2 Monte-Carlo methods

Equation 3.2 basically adds up layers of light that have bounced k times in the scene before reaching the camera. Monte-Carlo methods solve the rendering equation using this property. They express the light $L(p)$ that comes to a pixel p at the sensor as the integral of the contribution $C(p, l)$ of all possible light paths l between the light sources and the sensor:

$$L(p) = \int_{l \in \mathcal{L}} C(p, l) dl \tag{3.4}$$

In this expression $\mathcal{L}$ is the space of light paths, which dimension is also infinite. Monte-Carlo methods use the following—embarrassingly simple—algorithm to compute $L(p)$ for all pixels at once, repeating the following operations:

1. Draw a light path from the light sources to the camera with a probability proportional to the contribution of that path to the image;
2. sum the contribution to the pixel on the sensor that this path intersects.

In practice, sampling light paths proportionally to their contribution in the image needs to follow the light from the sources and bounce it on the geometry using importance sampling and the Russian roulette to decide whether the path is absorbed or not. That explains why Monte-Carlo methods, at least with such a naive implementation, converge very slowly.

Despite this fact, Monte-Carlo methods can simulate anything including complex light paths (think of e.g. sunlight creating caustics on the bottom of a swimming pool and refracting back to the camera, or dispersion effects in a diamond), motion blur, depth of field effects caused by finite aperture sensor systems, participating media, complex materials, etc. Depending on how light paths are sampled, some specific types of light paths can become much harder to sample in some methods, hence producing noise for particular subsets of the space of light paths.

Various advances in how to sample light paths in Equation 3.4 have given birth to a wide spectrum of methods, each having their strength and weaknesses [Dutré 2003]. I'm listing below a representative subset of three of them:

Path Tracing These methods randomly sample light paths from the source to the camera using various heuristics [Kajiya 1986]. The simplest path tracing approach consists in sampling from the source and rejecting paths when light is absorbed, until they reach the camera. When that occurs, a pixel corresponding to that light path is lit on the screen. Efficient approaches associate multiple sampling strategies, for instance connecting partial paths sampled from the camera and from the light, as with Bidirectional Path Tracing [Lafortune 1993]. Because they are absolutely generic and unbiased, path tracing methods are usually chosen for generating reference solutions.

Photon Mapping Photon mapping is a density estimation method [Jensen 1996]. The density that is thereby reconstructed is the distribution of radiance in the scene. It is estimated by measuring the amount of "photons" that accumulate in the scene. These photons are samples of the flux of radiance. In practice, the algorithm works in two steps. First, it generates light paths from the light sources and deposits photons at each bounce. Then a density estimation is performed for each pixel in the image to estimate the radiance at that pixel in direction to the camera. In their original formulation photon mapping methods are biased because the reconstruction kernel has a finite size. The recently introduced Progressive Photon Mapping removes this issue by constantly decreasing the size of the density estimation kernels while keeping the variance below a given threshold [Hachisuka 2009].

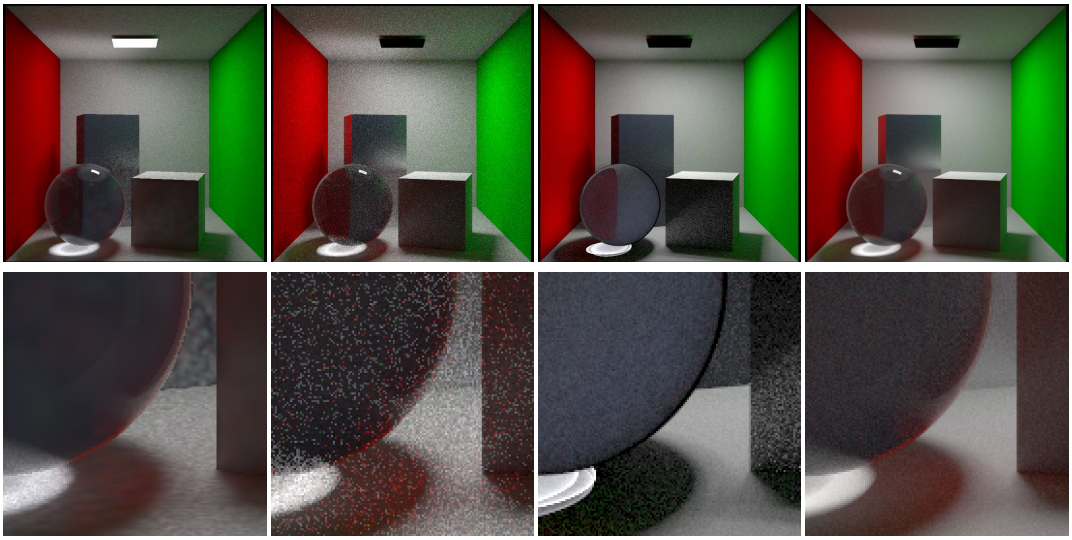
Metropolis light transport This method is an adaptation by Erik Veach of the Metropolis-Hasting algorithm for performing density estimation using a Markov chain of samples [Veach 1997]. The space considered is the infinite dimensional space of light paths that come trough the screen, and the density that is reconstructed is proportional to the amount of energy at each pixel in the image. The MLT method basically samples light paths and mutates them into similar light paths in path space, according to the ratio between their contribution to the image. Because of this, the MLT method is notoriously efficient for exploring regions of light path space that are difficult to sample. A common drawback is that the space of light paths is not uniformly sampled in the early stages of convergence. Instead, one can observe a strong correlation due to sampling some specific families of light paths (See Figure 3.2). MLT has been improved in various ways [Kelemen 2002, Hoberock 2010, Lehtinen 2013a].

Figure 3.2 shows the early convergence states of these three methods, along with a reference—almost—converged solution. This comparison puts the light on the various bias and variance effects one usually observes with these classes of methods, since the full convergence is only obtained after a significant amount of time.

Improving these Monte-Carlo methods has been relying on several different strategies, which just cannot be shortly summarized. Although each of my papers include very detailed previous work for the relevant techniques, it is possible to sort these strategies into high level sub-classes, among which are: the improvement of sampling so as to remove variance [Kirk 1994, Ashikhmin 2001]; noise removal during calculation [Pajot 2011] or as a post-process [Sen 2012]; exploit of coherency in the space of light paths [Keller 1997, Segovia 2006]; caching intermediate results during the computation in order to save path sampling [Jarosz 2008, Schwarzhaupt 2012], and clever reconstruction of the signal [Hachisuka 2008, Lehtinen 2011, Lehtinen 2013b].

A different way to look at this problem is to remark that Monte-Carlo methods work without priors about the solution that is being computed. Path tracing and Metropolis Light Transport for instance add up pixel-size light contributions to the final image in a way that is unaware of the structure of the image. Photon mapping performs density estimation possibly adapting the kernel size to the local density of photons whereas it should adapt it to the second derivatives of the energy (See for instance Silvermann's book, page 85 [Silverman 1986]). Looking at the images in Figure 3.2, it seems that if we are able to predict smoothness in some regions of the image before it is actually computed, we can afford to sample it less according to the Nyquist reconstruction theorem. Similarly, if we can predict the variance that will result from sampling a given subset of light paths, we might be able to increase the sampling rate accordingly.

Both variance and bandwidth are easily computed from the Fourier spectrum of the light field in a local region of path space. But this is a chicken and egg problem, since computing that Fourier spectrum requires to know the distribution of radiance



(a) Unconverged Photon mapping (b) Unconverged Path tracing (c) Unconverged Metropolis LT (d) Reference

Figure 3.2: *Comparison of the early seconds of convergence of the three classes of methods (a) photon mapping, (b) Bidirectional path tracing and (c) metropolis light transport, to the reference image in the classical Cornell Box scene (advancely converged Path Tracing). In this example one can easily spot how each method approximates the solution in its early stages of convergence: Photon mapping introduces density estimation bias and variance; Path tracing introduces high frequency variance proportionally to energy; Metropolis Light Transport forgets some entire regions of the space of light paths. Of course, all methods eventually produce the reference image on the right if allocated enough computation time.*

in the scene, which is what we are trying to compute in the first place! Our work on the frequency analysis of light transport is all about making this happen, and taking advantage of the level of smoothness of the illumination in light path space. I will present this work in more details in Section 3.2.

3.1.1.3 Remarks

To some extent, Galerkin and Monte-Carlo methods work in two ways dual to each other. While the former propagates light in the entire scene bounce after bounce, the later adds up contributions of single light paths that can bounce an arbitrary number of times. One big advantage of Monte-Carlo methods is that they do not rely on a mesh representation of the scene, making the computation really efficient in terms of memory cost. Galerkin methods however compute a solution that does not depend on the camera, which can be useful in some situations.

It is interesting to note that methods exist in-between these two concepts such as stochastic radiosity methods [Neumann 1994, Sbert 1998], but tend to combine the

limited ability to represent materials of the former and the low convergence speed of the later.

It is also known that the computation of visibility is the bottleneck of light simulation methods in general. This is illustrated by the fact that the computation of form factors is the bottleneck of the radiosity method, and that connecting light paths is the bottleneck of Monte-Carlo algorithms. Existing works provide extensive solutions for improving visibility queries [Havran 2001, Foley 2005, Hapala 2013].

3.1.2 Radiative exchanges in plant models

Despite their high memory cost and limited capability for computing global illumination in the presence of highly specular materials, radiosity methods offer the unique advantage of computing a solution that is independent of the viewpoint. They also have a simple formulation and an efficient solving algorithm for scenes that display purely Lambertian surfaces. Therefore, using the radiosity method for simulating radiant energy transfers in plant models is an appropriate choice. Some applications such as remote sensing require simulation in the near-infrared domain, where plants are highly diffusing reflectors with an albedo close to unity.

My contributions in this field has been to develop radiosity methods that are suitable for the challenging case of lighting simulation in vegetation cover, and to experiment the application of these methods into two different fields: physiological plant-growth simulation and remote sensing.

3.1.2.1 Instantiation for lighting simulation in plant models

A first step in this work has been to design the principles of using the radiosity method in combination with geometric instancing. This is not as easy as it sounds, since traditional hierarchical radiosity methods are by design attached to an explicit representation of the underlying geometry. The algorithm we proposed extends the hierarchical radiosity algorithm of Smits *et al.* [Smits 1994] that groups objects into clusters between which energy exchange can be represented by a single form factor.

Our contribution was to prove that it is possible to treat instanced objects in the hierarchical radiosity method as black boxes in the hierarchy of clusters, for which a directional transmittance and a phase function need to be pre-computed. Once an equilibrium of the radiant energy has been found at the high level of these black boxes, the solution is refined in each cluster, sequentially replacing each black box by its actual geometry, possibly made of smaller instances. Each opened cluster is treated by considering external incoming energy as constant, since it is already the result of the converged solution at upper hierarchical levels. The method recursively treats all instances in the scene, while sequentially outputting the solution on the disk. This algorithm is general enough to apply to any scene with instantiation.

Later on, we studied the application of this algorithm to the specific case of plant models. That involved finding a heuristic to determine how to create an instantiable hierarchy of clusters in a plant model and properly precomputing and

storing radiometric information (e.g. phase and transmittance functions) of plant models **[RI-10]**.

3.1.2.2 Plant growth simulation

In a collaboration between INRIA and the CIRAD (French Agricultural Research for Development) and LIAMA (French-Chinese Laboratory for Automation and Applied Mathematics, in Beijing), we have brought together the simulation of radiative exchange in plant models and the simulation of plant growth.

The system works as an interaction loop between two software components: (1) The plant growth simulator computes the shape of the plant at the next time step, given the amount of radiant energy received at each leaf. The plant growth simulation model that is used, is the *GreenLab* model, originally developed by Philippe Dereffye. This simulator combines measured physiological data on one side, and a plant growth model that has been validated by measurements on real plants in a controlled environment [Cournède 2009]; (2) The light simulation engine takes as input the geometry of the plants at the current step and computes the amount of radiant energy received by each leaf. We achieved that using a modified version of the software developed in collaboration with F.Sillion and G.Drettakis, which has been named PlantRad **[RI-10]**.

We have experimented in various situations where plant growth is influenced by the distribution of light. Trees growing at a close distance will mainly develop new leaves near the top; plants will preferably grow toward light sources (heliotropism); Most importantly, these physical modifications of the plant models are not the result of arbitrary rules when shaping the plant, but the reaction of a physiologically callibrated model to simulated environmental factors. Figure 3.3 shows such an example where a small plant grows toward the light and finally out of the window.

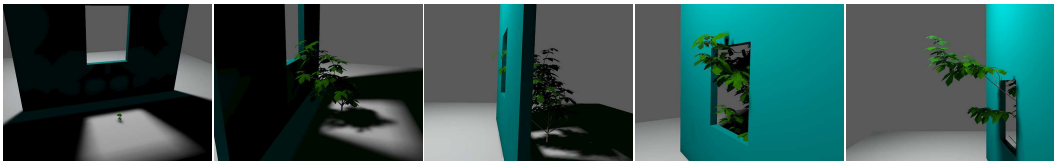


Figure 3.3: *Simulation of plant growth that results from the interaction between a physiological plant simulator and a simulation of radiative exchanges. This model generates realistic plant models that adapt to the geometry and lighting conditions of their environment.*

3.1.2.3 Application to remote sensing

The shape and intensity of the reflectance function of lands covered by forest gives important clues about the plant species involved. Measurements are typically performed using the sun as a light source, and captured from a satellite. The resulting

reflectance function is a complex combination of the direct illumination from the sun, diffusion in the sky and possibly intricate interaction inside the foliage. A typical pattern that is observed is the angular width of the *hot-spot*, which is the region where self-occlusion of the forest becomes negligible because the directions of the light source and the camera are very close.

Simulating the reflectance that results from such a complex system is extremely challenging. The large amount of geometry involved limits the possibility of an accurate calculation in a reasonable amount of time (typical test scenes involve thousands of trees having hundred thousands of leaves each). The various approximations involved in this calculation (simplified geometric models for trees, simplified reflectance model, simplified atmospheric model, etc) have an impact on the final reflectance estimate that remains very hard to quantify. Moreover, while trees reasonably block the light in the visible spectrum, they do behave light almost pure diffusing volumes in the infrared range, making light simulation very long to converge.

For these reasons a RAdiation transfer Model Inter comparison (RAMI) is organized every 2-3 years [2], and scientists can participate and compare the results of their simulations over a set of standard scenes.

We participated in phase III of RAMI **[RI-06]**, providing measurements using our radiosity engine with instantiation. This was done in collaboration with Noveltis, a company in Toulouse that performs technology transfer for the industry [3].

3.2 Fourier Analysis of Light Transport

As explained in Section 3.1.1.2, Monte-Carlo methods for global illumination basically sum up contributions of light paths, that are randomly sampled with various strategies from the space of all light paths between the sensor and the light sources in the scene. We saw that a possible way to improve these methods is to perform a predictive analysis of the bandwidth of the illumination in light path space. In this chapter, I give a comprehensive high-level overview of the Fourier analysis of light transport and its applications. A more technical and rigorous description is provided by each of the publications I participated to **[RI-00,RI-01,RI-02,RI-05,RI-08,CI-02,PI-01]**. Whereas the original 2005 paper establishes the fundamentals of the theory **[RI-08]**, later contributions explore its possible extensions to specific lighting simulation problems and provide practical applications.

This work has been the conjoint effort of many contributors, among which Fredo Durand (who had the original idea), Nicolas Holzschuch, Kartic Subr, Laurent Belcour, Kavita Bala, and Mahdi Bagher.

Local light fields in path space The Fourier analysis of light transport is based on the notion of *local light fields*, which represents the 4D subset of rays that lie in

[2] See `http://rami-benchmark.jrc.ec.europa.eu/HTML/`

[3] See `http://www.noveltis.com/`

the neighborhood of a given light ray. To represent such a set, we use the paraxial parameterization of rays based on two Cartesian coordinates δu and δv in the plane orthogonal to the ray and two angular coordinates $\delta\theta$ and $\delta\phi$ as shown in Figure 3.4.

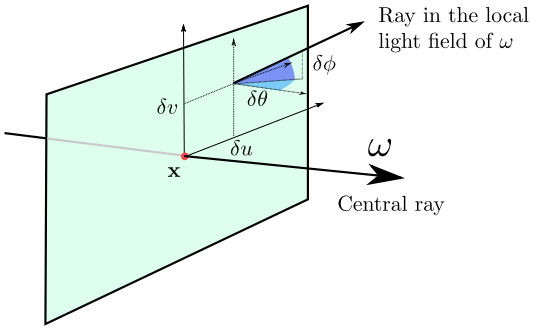


Figure 3.4: *Paraxial parameterization of rays in the neighborhood of a light ray.*

For a complete light path, one can concatenate the local light fields around segments of the path between successive bounces.

The notion of "locality" plays an important role in the definition of local light fields, and seems rather arbitrary at first. For the theory to be useful and sound, the local light field needs to be the largest region around the central ray in which the signal can be seen as stationary, or in other words, when its variations can be represented by the intensity spectrum of the signal. In practice, the exact "width" of the local light field never explicitly occurs in the calculations, except for defining visibility events, and the central ray can be considered a representative of what happens in its neighborhood.

Transport operators in the Fourier domain The original 2005 paper demonstrates that the various events a light ray and its local light field encounter all turn out to be simple operators in the Fourier domain **[RI-08]**. In particular, free space traveling causes a re-parameterization and therefore a shear between spacial and angular coordinates, which is also a shear in the Fourier domain. Reflectance causes a convolution in the primal domain, which in Fourier space is a simple product between the light field and BRDFs spectra. Partial visibility is a product with the binary visibility function in the primal domain, and therefore a convolution in Fourier space.

As a consequence, knowing the Fourier spectrum of the local light field at a point leaving a light source, one can easily predict approximately what the spectrum of the light field will be after several interactions with the environment.

This high level of simplicity totally justifies the use of the Fourier theory in place of wavelets or polynomial functions to characterize the light transport operators. Indeed, only the Fourier basis brings at once the convolution theorem (that turns convolutions into products) and linearity with respect to transforms in the 4D space (for instance the Fourier transform of a rotated signal is a rotated copy of its Fourier

transform). These good properties all come at the cost of locality. But because we're dealing with stationary signals, this is not a problem *per se*.

An essential contribution of the Fourier analysis of Light Transport has been to measure local light fields (using ray-tracing) and effectively compute their Fourier spectrum in order to verify that the predictions given by the theory actually matches the measurements. An example of such a measurement is given in Figure 3.5 below:

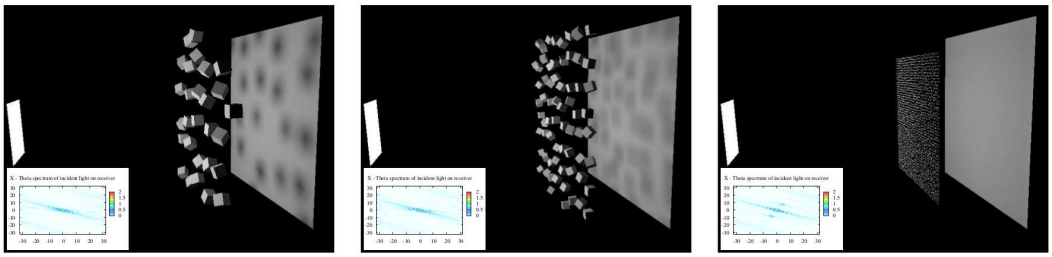


Figure 3.5: *Practical measurement of the Fourier spectrum of the light field just before hitting the screen on the right. In all three cases, the spectrum of the light field carries multiple copies of itself, as a result of the convolution with the spectrum of the set of blocking cubes. However, if the cubes are small enough, these copies spread far from the main (tilted) spectrum, and after reflecting on the diffuse reflector, which retains only the spatial component of the spectrum (the horizontal central axis of the spectrum here), all frequencies will disappear. This is consistent with the absence of shadows at right, where the blocker's spectrum only has high frequencies.*

Using such an explicit definition however, the Fourier spectrum of a local light field along a light path might need a tremendous computational effort to compute, which certainly cannot be afforded for all light paths in a Monte-Carlo simulation. Sampling a local light field using as few as 128 samples along each of the 4 dimensions of the space for instance, already requires $4 \times 128^4 Bytes = 1GB$ of storage. Consequently, we need a method to efficiently compute or predict the spectrum, and we need a compact representation of that spectrum.

3.2.1 The covariance representation

We came up with an efficient representation for the spectrum by examining which are the characteristics of that spectrum we might potentially be interested in. In order to define a proper sampling density for image reconstruction, one needs to know the combined bandwidth of the signal along all axis of the data in order to follow the Nyquist reconstruction theorem. If we want to design proper reconstruction filters in the image, we need to know whether the signal is anisotropic (meaning it has a preferred direction) and in which direction of the space this signal is elongated/stretched. All these characteristics are exactly represented by the covariance matrix of the power spectrum of the signal, that is defined by:

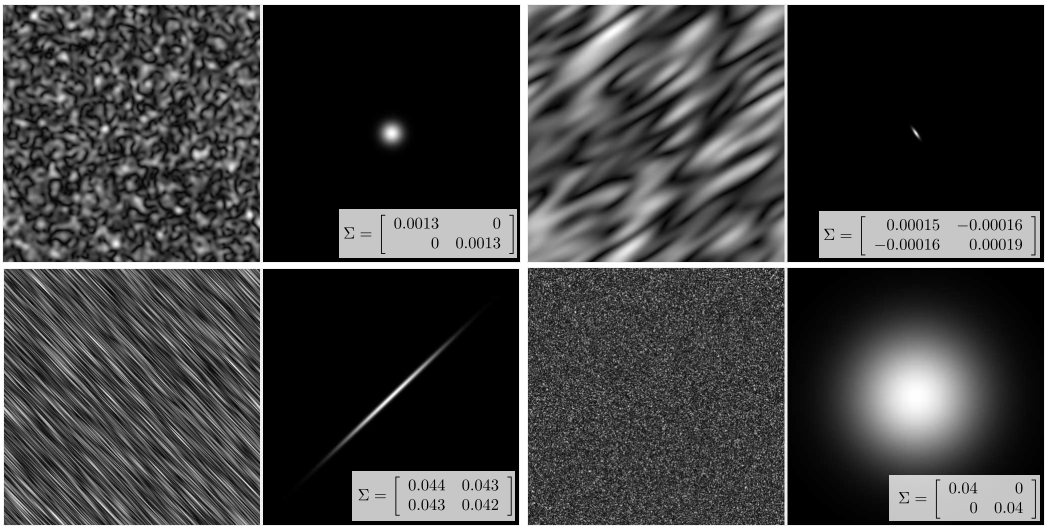


Figure 3.6: *Practical examples of stationary signals, along with their power spectrum. These examples where generated by inserting random phases into a Gaussian power spectrum. For each image, the covariance of the 2D power spectrum is shown.*

$$\Sigma_{ij} = \int_{\mathbf{z}\in\Omega} < \mathbf{z}, \mathbf{e}_i >< \mathbf{z}, \mathbf{e}_j > |\hat{l}(\mathbf{z})| d\mathbf{z}$$

In this equation, Ω is the 4D Fourier space, and $\mathbf{e}_i$ is the i^{th} canonical basis vector of that space. The traditional definition of the covariance of positive functions usually involve the expected value of that function. Because the signals we're dealing with are real, their Fourier spectrum is symmetric and therefore the expected value of a probability density function proportional to $\hat{l}(\mathbf{z})$ is always 0. The covariance matrix also offers very interesting properties: the covariance of the signal is exactly the inverse of the covariance matrix of its Fourier transform and the covariance operator is additive. Therefore it seems very reasonable to adopt the covariance of the Fourier spectrum of the lightfield as a compact representation of that spectrum. This is illustrated in Figure 3.6 where I show different stationary signals that have been generated from a Gaussian power spectrum with random phases. Small values mean low bandwidth and therefore large oscillations in the signal. The eigenanalysis of the matrix also reveals the orient of the spectrum and therefore the anisotropy directions of the signal. This proves the high relevance of the covariance representation.

Recomputing the covariance matrix of the Fourier spectrum at each bounce along a light path sounds like a terribly inefficient task. Fortunately, we found the perfect way to do it: it is possible to directly derive matrix operators that transform the covariance matrix of the Fourier spectrum of local light fields for each of the light transport operators: free-space transport, occlusion and reflectance. In practice, that means we only need to be able to compute the Fourier covariance at the emit-

ting light sources, and then transform these covariances when the light is reflected, transported, and occluded, using the appropriate matrix operators. Eventually, the calculation that is required in order to know the covariance of the Fourier spectrum of the light field in the camera sensor is a bunch of matrix operations, which is certainly practical for applications in path tracing, and possibly real-time rendering.

Representing the spectrum of the local light field by its covariance is exactly equivalent to approximating that spectrum by an anisotropic Gaussian. Indeed, the covariance matrix Σ of $\hat{l}$ is also the covariance matrix of the Gaussian defined by:

$$g(\mathbf{z}) = e^{-^T\mathbf{z}\Sigma^{-1}\mathbf{z}}$$

Such a representation however is not convenient for expressing the light transport operators in the Fourier domain, because some of these operators zero the covariance along particular directions by turning the spectrum into a Dirac along these directions, making Σ non invertible. Nevertheless, we showed that in this case the expected operators can be expressed using the pseudo-inverse of the covariance matrix, as it happens for the reflectance operator.

3.2.2 Application to depth of field and motion blur

Depth of field and motion blur are two notable effects that "blur" the content of an image in different ways. The former is the result of an image focusing in front or behind the camera sensor because of the optics of the finite aperture camera. The later is the result of the non instantaneous capture of moving scenes due to the limited sensitivity of camera sensors, which effectively blurs the captured image in the apparent direction of the movement. Figure 3.7 shows an illustration of this.

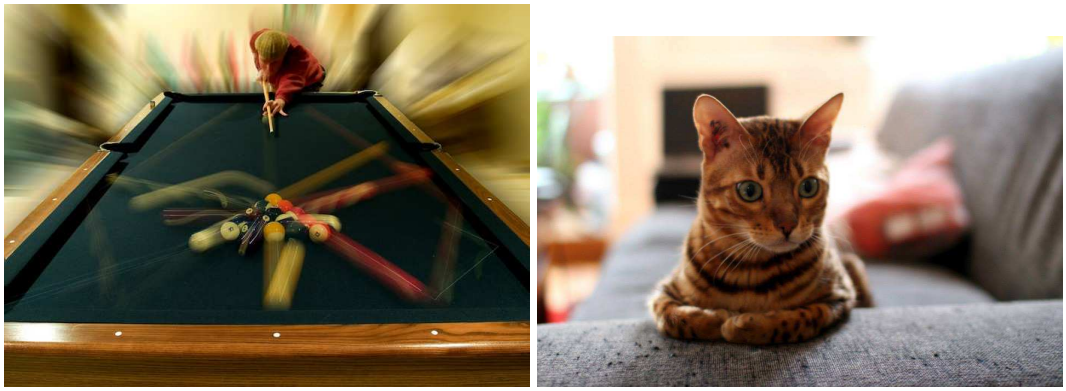


Figure 3.7: *Extreme motion blur (at left) and depth of field effects (at right). While these effects blur out information in an image, they require extra computational power when simulated using Monte-Carlo methods. Our contribution of Fourier analysis of light transport predicts and takes advantage of the reduction of information in the image to reduce the required level of computation.*

A common characteristic of these two phenomena is that Monte-Carlo algorithms, while unbiased, spend a large computation time performing the integration

across the camera lens and shutter time interval, in order to produce an image that is eventually very smooth. Indeed, because this integration practically averages the signal, it will heavily remove high frequencies and therefore reduce the amount of information in it. Computing motion blur and depth of field images efficiently was therefore a good candidate application for the Fourier analysis of light transport.

In a first step, we have performed a frequency analysis of the finite aperture camera model **[RI05]**. We show that the camera acts as a band-limiting operator in the angular domain, effectively cutting off high frequencies of the signal. In this work, the Fourier analysis is performed for the last bounce before the camera, and conservatively supposes infinite spatial bandwidth in the signal leaving the objects in the scene. The model was validated by comparing the predicted bandwidth to the bandwidth of windowed Fourier transforms of light fields obtained by practical measurements in image space. Finally, we derived an *ad hoc* image reconstruction method based on splatting isotropic Gaussian kernels, which size is computed from the frequency estimate of the signal in image space.

While this work effectively brought a method for computing images with depth of field that outperformed blind ray-casting, it suffered from several limitations:

- the whole analysis was performed in 2D (one angular and one spatial dimension), making it impossible to predict anisotropy in the signal and use anisotropic reconstruction kernels;
- the internal representation of the power spectrum of the light field was based on a collection of 2D samples, making it quite inefficient to compute.

To overcome these problems, we have worked on designing a more efficient representation of the spectrum: the covariance matrix **[RI01]**. As compared to the sampled representation of the power spectrum, the covariance representation offers

- a full 5D anisotropic analysis in the frequency domain (2D space, 2D angle, 1D time);
- a very efficient computation of the effect of light transport operators in the frequency domain;
- a general analysis that applies to any light path, potentially useful for all Monte-Carlo methods.

Moreover, we have extended our frequency analysis to time-dependent signals, making it possible to represent time shears caused by moving objects or various camera motions. This allowed us to correctly predict anisotropic reconstruction filters for images with motion blur. Figure 3.8 below shows multiple examples of predicted covariance matrices in a scene displaying both motion blur and depth of field effect, with specular, glossy, and diffuse materials.

Because the 5D covariance formulation allows a full anisotropic analysis of the signal, we are able to derive reconstruction filters in screen-space that are correctly aligned with the directions of anisotropy in the screen.

Figure 3.8: *5D covariance matrices of local light field predicted by the frequency analysis of light transport. The moving white ball is in focus, but blurred because of the relatively long aperture time. Therefore it shows a strong correlation between time and the four other dimensions.*

3.2.3 Fourier analysis for density estimation methods

As explained in 3.1.1.2 density estimation methods perform the reconstruction of an unknown density from a random set of samples issued from that distribution.

It has been shown [Silverman 1986] that the error in the reconstructed density is the result of two terms: a variance term, that is the result of too small a reconstruction kernel, which is proportional to the estimated density, and a bias term, that is the consequence of using a reconstruction kernel that is too large with respect to how far from its average the density is. That later term is proportional to the Laplacian of the function. The final error measure, proposed by Silvermann is:

$$E(\mathbf{x}) = \alpha h^2 \triangle f(\mathbf{x}) + \frac{\beta}{nh^d} f(\mathbf{x})$$

In this expression both α and β are constants that depend on the shape of the reconstruction kernel, f is the recontructed signal, d is the dimension of the space. The optimal size of a reconstruction kernel h is what minimizes this error for a given number of samples n.

We have shown that the eigenvalues of the covariance of the spectrum of the local light field around a point $\mathbf{x}$ are equal to the eigenvalues of the Hessian matrix of that function, up to the sign (See **[RI-00]**). This allowed us to derive a conservative

approximation of the bias error in the above equation, as a function of the predicted covariance.

In practice, this brings very useful information for global illumination methods based on density estimation, for instance photon mapping [Jensen 1996], and progressive photon mapping [Hachisuka 2009]. In the former, computing the covariance matrix of the light field in the Fourier domain allows to know, for each photon, what would be the optimal reconstruction kernel.

In the method of progressive photon mapping, reconstruction kernels are carefully reduced to zero, in order to reduce the bias while maintaining variance below a given threshold. Thanks to our prediction, we know that reducing them is not necessary as soon as their size is small enough to ensure an acceptable bias. In practice, this means that we can eventually keep the same radius as the number of samples increases. Because of this, variance reduces and our solutions converge faster at equal error **[PI-01]**. We also applied this to accelerate Progressive Photon Beams **[RI-00]**.

3.2.4 Fourier analysis of participating media

Participating media designate volumetric media that contain reflecting particles which size is much smaller than the wavelength of light (for instance fog and translucent objects). As a consequence, participating media act on light as a diffusing medium. The light equilibrium equation in participating media extends Equation 3.1 with two additional terms: an absorption term and a scattering term (The reflectance term on objects is omitted for clarity):

$$\omega.\nabla_{\mathbf{x}} l(\mathbf{x},\omega) = \kappa_a(\mathbf{x})(l_e(\mathbf{x},\omega) - l(\mathbf{x},\omega)) + \kappa_s(\mathbf{x}) \left(\frac{1}{4\pi} \int_{\omega' \in S^2} \rho(\omega,\omega') l(\mathbf{x},\omega') d\omega' - l(\mathbf{x},\omega) \right) \tag{3.5}$$

This equation basically says that the increase of radiance at $\mathbf{x}$ in direction ω is proportional to the emitted radiance minus the radiance that is absorbed, plus the radiance that is scattered inward from all incoming directions, minus the radiance that is out-scattered from direction ω.

If the case of a medium for which the mean free path of light is relatively small as compared to the domain, the solution of the scattering equation can be approximated by solving a diffusion equation [Stam 1995].

Another notable family of methods for solving this equation are based on a global Fourier analysis of the domain, which decorrelates the various frequencies in the solution. The equation in the Fourier domain can then be solved analytically [Ishimaru 1997]. Such methods however require strong assumptions about the domain and the phase functions as well, which do not realistically apply when computing global illumination for image synthesis. In this case, we're bound to using the Monte-Carlo methods which are general enough to handle all situations.

Because of their continuous and inherently 3-dimensional nature, participating media are the worst possible test bed for Monte-Carlo simulation methods. Indeed,

Monte-Carlo methods traditionally compute light propagation by summing contributions of a large number of independent light paths that randomly bounce in the medium. These methods eventually reconstruct an image by adding totally incoherent pixel-size contributions into an image that is otherwise smooth in many ways (See Figure 3.9). Consequently using frequency analysis of light transport in order to improve Monte-Carlo methods seems very relevant.

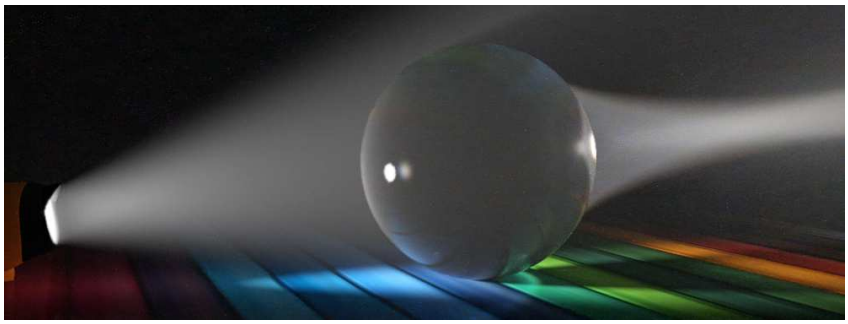


Figure 3.9: *Picture rendered with the ARION™ physically based renderer to illustrate the complex effect of participating media. High frequencies created by light sources, occluders and caustics get gradually blurred as the light diffuses inside the medium. Predicting the actual frequency content of light fields in this kind of scene is a real challenge.*

Our contribution in this work is 4-fold: first, we have performed an analysis of scattering and absorption in the Fourier domain. This means that starting from Equation 3.5, we have derived the mathematical relationship that express the behavior of the local light field along a light path that travels in participating media. This part of the work proves that scattering acts as reflexion and lowers frequencies, whereas absorption acts as a non binary visibility filter and therefore increases bandwidth. The later sounds counter-intuitive, but the effect is clearly visible on "god rays" created by clouds, or the shadow cast by the glass sphere in Figure 3.9.

Secondly, we have derived the corresponding operators for the covariance matrix of the power spectrum. That includes finding a practical way to compute the covariance of phase functions in the 4D chunk corresponding to the local lightfield at a scattering event, and computing the frequency content of the absorbance.

Third, we observed that the light-path based analysis cannot be used directly to drive sampling and reconstruction since the bandwidth in the image (and in the volume) eventually results from the combination of many light paths. So we figured out methods to convert the per-light path information into practical sampling metrics. To do that, we introduce an intermediate quantity, called the volumetric covariance, that represents the spectral covariance of the fluence in the volume. Using this quantity, that we obtain by summing slices of 4D covariance contributions from many rays into a 3D grid, we show that it is possible to compute a number of very interesting metrics: the variance of the fluence along rays from the camera, the Laplacian of the fluence in the volume, and the covariance of the 4D radiance

field in screen-space.

Finally, we use these metrics to improve existing computation methods. We propose four scenarios where our frequency analysis allows a faster computation: we design an adaptive sampling for integration of light toward the camera; we use the Laplacian of the fluence to compute an optimal radius reduction strategy for the method of progressive photon beams; we use screen-space covariance to derive optimal image reconstruction filters; finally, we show how the Laplacian of the fluence can be used to improve irradiance caching methods **[RI-00]**.

3.3 Real time rendering

Real time rendering is devoted to porting existing algorithms to the Graphic Processing Units (GPU), or develop new techniques to perform existing tasks using graphics hardware. GPUs have grown in power at a speed larger than what is predicted by Moore's law. At the time of writing this document, a 300$ graphics card has a power of more than one Teraflop, which for comparison is the computational power of the *Cray 2* supercomputer in 1990. The ability of researchers in computer graphics to use such a low cost and powerful hardware has brought an entire scope of possible research.

Now, and since approximately 10 years, GPUs are programmable, with a shader language that has evolved to closely reassemble C++. In addition to supplying graphic primitives to be rendered, the user can also supply shader code that will be applied in parallel to these geometric primitives, and to pixels in the screen, as illustrated in Figure 3.10. It has therefore become quite common to use GPUs as an efficient programmable test bed for processing not only graphical primitives, but also any numerical data, by hacking the graphic pipeline for tasks that would not necessarily aim at producing images. This was called GPGPU computing. More recently, NVidia has released a GPU programming language that is not tied to the graphic pipeline anymore, called *cuda* [Sanders 2010] which allows to program graphic cards without the constraints imposed by the data layout of the graphic pipeline.

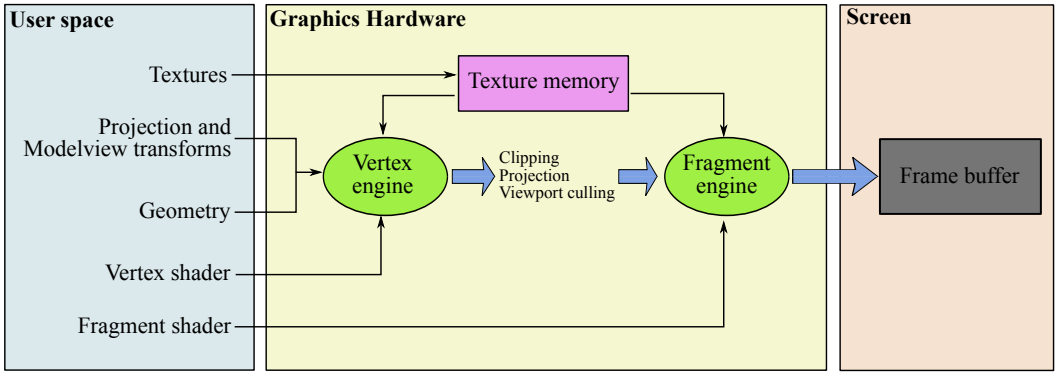


Figure 3.10: *Simplified view of the graphics pipeline. Items in green are programmable with C-style code, which offers enormous possibilities of using graphics cards to compute the rendered result on the fly. The user only interacts by changing the data in the "user space" area of the pipeline.*

But because of their very specific architecture, GPUs need friendly algorithms, and although the shader language looks like C++, it imposes some specific algorithmic requirements which mainly reside on two characteristics: (1) memory accesses are very costly and should therefore be considered as a bottleneck. Memory accesses are facilitated by a hierarchy of caches based on locality in the images, which forces

the programmer to keep memory accesses local in texture or pixel space; (2) the implicit operating mode is SIMD which stands for Single Instruction on Multiple Data. That needs a major rewrite of algorithms. Because of their SIMD architectures, graphic cards do not like algorithmic branching at all. Indeed, when a warp of graphics processors have to split between different branches of a program because the underlying data needs it, twice the time will be used since both branches are executed sequentially. As a consequence, porting existing algorithms to GPUs is generally considered to be a research problem.

In this chapter I present three projects that share the same scientific approach: they aim at simplifying the equations of a known physical phenomenon in a controlled manner, in order to produce a GPU-friendly algorithm. The goal is to produce real time results with controllable error, while taking into account the limitations of graphics hardware programming, essentially the SIMD model and the memory access bottleneck. Of course, because real-time rendering algorithms highly depend on the capabilities of GPUs which evolve really fast, it is likely that the research that is done in this context has a reasonably short life time. The value that remains however is a better understanding of the physical and algorithmic aspects of the different problems in the light of image synthesis.

3.3.1 Revisiting birefringent materials

This project aimed at rendering facetted gemstones in real time, using graphics hardware. Facetted gemstones take their colors from two optical phenomena:

Polarization While light polarization might safely be neglected for image synthesis in most applications, this is not true for gemstones because of the phenomenon of *pleochroism*. The absorbance of light in gemstones depends on its polarization. The optical characteristics of the Fresnel reflectance at the interface between the material and the air strongly affect the polarization of light. As a consequence even unpolarized light entering a gemstone becomes polarized after encountering its surface and its polarization changes when light bounces inside the stone. Figure 3.11 below shows an typical example of this phenomenon on a photograph of a tourmaline.

Dispersion That effect is the consequence of the refractive indices not being equal for all wavelengths across the spectrum of visible light. The primary effect is that white light entering a gemstone is spread into a continuous range of colors with different directions of propagation. This phenomenon is responsible for the fire of diamonds, as well as most stones with a moderately high refractive index.

As often in nature, these characteristics can be found in all possible combinations at various scales. Examples of non birefringent stones include diamond and red garnet, birefringent stones include sapphire (uni-axial) and andalusite (bi-axial). In summary rendering gemstones is a pain since it needs quite complex calculations

for estimating the transfer coefficients at each interface. As a consequence, accurate renderings of such objects in real time was a real challenge.

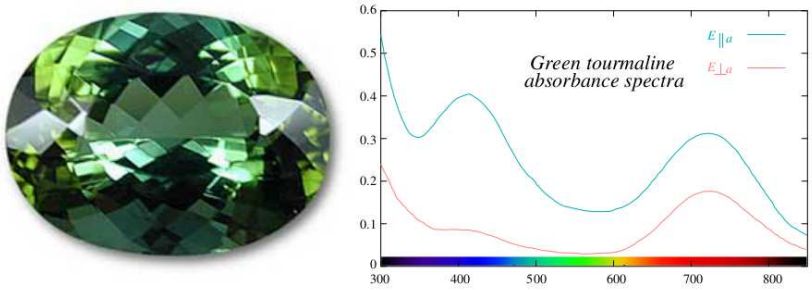


Figure 3.11: Typical display of the effect of pleochroism in a tourmaline gemstone. Light traveling horizontally is not absorbed the same way than light travelling vertically (w.r.t. the observer) which causes this apparent difference in color within an material that is otherwise optically constant.

Realistically rendering faceted gemstones in real time has a few potential applications the first of which is jewelry prototyping: vendors would be able to foresee the look of a jewelry project before even building it. A gemstone rendering system that would allow to interactively change all parameters would also have a high educational impact. Finally optimizing gemstone shapes to produce new interesting cuts is an interesting application, although a very accurate rendering (in both its spectral and geometric aspects) is necessary to confirm what a real time simulation can produce. Our contribution to this topic has been three fold:

First of all, we have gathered the equations and optical laws to be used in the context of ray tracing, which is harder than it seems, because optical laws are not always presented in books in a form that is consistent with ray-tracing. Eventually, we had to re-derive everything from scratch. But because of this, we were able to give a complete set of equations and a precise road map for someone to implement ray-tracing in birefringent materials.

The second contribution was to simplify the optics of birefringent materials to make it tractable on GPU. In 2004 indeed, graphics hardware was far less powerful than it is now. That means quantifying the visual importance of each phenomena. Among questions to be solved are for instance: to what extent is a linear approximation of the absorption acceptable, or should we account for the doubling of the edges seen after internal reflections? Other examples include using a proper color space, or limiting the depth of the internal bounces of light in the minerals.

Finally, we designed an algorithm to render gemstones on the GPU. For this we used the *feedback buffer* to clip polygons and therefore determine regions that delimit optical paths with similar parameters. Each region was then rendered in high dynamic range using a shader that would account for the various light transport events such as internal reflexions, changes in polarization and absorption, which we completed by a tone mapping pass **[RI-09]**.

The prototype we designed had the interesting property to allow us to modify any parameter of the rendering in real time, including the stone's geometry, absorption parameters, index of refraction and direction of the optical axis.

Validating our results was the most challenging task. First, we implemented a ray-tracer to serve as a both a proof-of-concept for the derived equations and as a reference computation to validate the GPU approximations. We also performed comparisons to real images. This is obviously difficult, since the pixelwise appearance of gemstones is known to vary a lot with the orientation of the incident lighting.

Of course the GPU implementation which resulted from this work has rapidly become outdated. What essentially remains of this work is the overall recipe to ray-trace birefringent materials. If I would be to implement a new prototype for the same paper now, I would do it in cuda [Sanders 2010], and include possible improvements to this work such as rendering in full spectral mode, to avoid possible banding artifacts, and also generate caustics caused by light going out of the stone.

3.3.2 Screen-space indirect illumination

This project aimed at computing indirect illumination in screen-space for video games. It took part into a joint 'region' project between INRIA and video game companies including Eden Games, Krysalide and Gamr7, which ultimate goal was to create a fully working video game production pipeline. I was responsible for the scientific management of the "lighting" and its GPU implementation.

The time constraints for an algorithm to fit in a video game pipeline are drastic. Indeed, it is common in the research community to sell new algorithms as potential candidates for taking place into a video game pipeline when these algorithms can run at 30 frames per second (or 33 ms per frame) for moderately complex geometry. In reality, the limit of 33 ms per frame should already account for all other tasks required by the game including artificial intelligence, physically based animation, rendering of all geometric primitives (a large number of them sometimes), disk accesses and memory management. That means in particular that a global illumination algorithm that would takes place at the end of the graphics pipeline should not be using more than 3-4 milliseconds per frame, whatever the amount and variations in complexity of the geometry of the scene. Consequently, our primary line of work was to simplify the computation of indirect illumination to its bare minimum and keep it as independent as possible from the geometric complexity of the scene.

In order to handle this, we decided to perform the whole calculation in screen-space. This also favored the integration of our algorithm in the deferred shading pipeline that was used at Eden Games. We first rendered full screen-images of the scene with normals, materials and depth. Then a shader taking these images as input gathers illumination from the entire scene at each point in the image. Obviously, only parts of the scene that are visible can be used to gather light from. That is a severe approximation, which we compensated by merging the indirect illumination using multiple cameras **[CN-04]**.

In this algorithm, texture accesses is the main bottle neck. In particular, illumi-

nation is potentially exchanged between any pair of points in the screen. Therefore we designed a hierarchical algorithm to limit distance in texture accesses: all input data is mipmapped with proper algorithms, and light is gathered from a ring of pixels in each mipmap. Eventually, when all levels are combined, light from all distances in the scene is gathered at each pixel. We also derived the equations for computing illumination in screen space without bias. An other important contribution was to handle temporal coherence, by merging several warped views when moving the camera, and to handle visibility between samples, using ray tracing into a voxel grid to represent the scene, which was computed on the fly using existing techniques [Eisemann 2006].

The algorithm was successfully tested in the engine of the game "Alone in the Dark", although not used in the final product that was already finalized. Figure 3.12 shows an example of employing our screen-space indirect illumination in the rendering engine of that game.

Direct illumination only

With screen-space indirect illumination added by our technique

Figure 3.12: Our method to compute screen-space indirect illumination was successfully used in the pipeline of the game *Alone In The Dark*, developed by Eden Games.

3.3.3 Rendering measured materials in real time

Visual realism in image synthesis essentially depends on the accuracy at which materials are represented[4]. Eventually, what we see in a synthetic image is reflected light, that always carries a very subtle fingerprint of the material where it has last bounced.

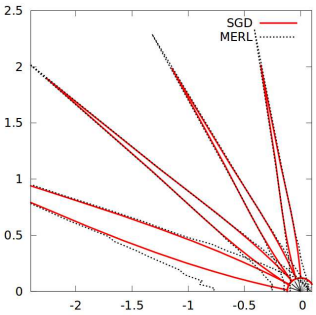
Yet, when a material's reflectance is convolved with natural light, the eye is immediately capable of categorizing the appearance into large classes (e.g. plastic, paint, metal, ...), and subclasses (brushed metal, unpolished metal,...), then eventually putting a name on it (gold, brass, silver, etc). Although the perceptual

[4]I do not have references to validate this claim; I think we should conduct perceptual experiments so as to measure the sensitivity of perceived realism for various approximations of material models.

dimension of this extraordinary capability does not seem to have been studied yet in the context of rendering, it seems that using a more or less approximate representation for the same materials will be immediately visible to a human observer. Figure 3.13 below shows the reflectance lobe of a metallic paint from the MERL database [Matusik 2003], which was acquired with a gonioreflectometer. Although an approximation of the measured BRDF with a Shifted Gaussian Distribution **[RI-03]** is numerically very close to the actual material (it is considered the best as of year 2013), the rendering with natural light still fails to convey the exact nature of the material.

Rendered with measured data

Rendered with best known fit (SGD)



Comparison of SGD and original models

Figure 3.13: *Even a reasonably accurate analytical approximation of a measured material is not always good enough to render the perceptual aspect of a material. Here, a SGD distribution (considered to be the best fit in 2013) does not fully convey the metallic aspect of this* green metallic paint *material from the MERL database.*

As a consequence, using an accurately sampled representation for a material's reflectance is mandatory for a number of applications, such as virtual prototyping of car paints. In practice, the BRDF will be convolved at each pixel with the incident illumination using the shading equation:

$$L(\mathbf{x}, \omega) = \int_{\Omega} \rho(\mathbf{x}, \omega, \omega') E(\omega') \cos\theta d\omega'$$

Numerically, this integral can be approximated using Monte-Carlo integration. A large collection of directions ω_i' is sampled according to a spherical probability density function p, and the sum is approximated by:

$$L(\mathbf{x}, \omega) \approx \frac{\pi}{N} \sum_{i=1}^{N} \frac{E(\omega_i')\rho(\mathbf{x}, \omega, \omega_i') \cos\theta_i}{p(\omega_i)}$$

When N grows to infinity, the sum converges toward the actual value of the integral.

However, computing this equation for all pixels is absolutely impossible to achieve in real time for multiple reasons: First of all, the convergence rate of Monte-Carlo integration is $\mathcal{O}(1/\sqrt{N})$. It can be raised to $\mathcal{O}(1/N)$ using stratification and further improved with importance sampling using a well tuned distribution for p,

but that still means that hundreds of samples of incident illumination need to be used for each pixel. Because memory look-ups have a very large cost on the GPU, no practical solution exist for computing this integral in real time for all pixels in an image with an acceptable amount of noise. We have of course explored alternative solutions for instance based on projection on directional function bases [Wang 2009]. But because we wanted to render the measured material instead of an approximation of that material, we eventually discarded these solutions.

Observing synthetic images of measured materials (see e.g. Figure 3.13), one immediately sees however that glossy and moderately diffuse materials tend to produce very smooth images. Glossy materials also produce smooth images if the illumination is smooth. Finally, the surface geometry, as well as the distance to the camera seem to take a strong part in influencing image-space frequencies in the final result. In practice, a subtle combination of all these parameters governs the pixel to pixel complexity of the resulting image. As a consequence, it appeared to be an interesting contribution to predict image smoothness at each pixel and adapt the computational effort to the least amount that is necessary to render the image correctly.

Our contributions have been to simplify the theory of frequency analysis of light transport for the simple case of shading from distant illumination, so as to perform frequency predictions directly on the GPU, and allow an optimal balance between accuracy and computation cost. We designed a multiresolution shading algorithm based on a one-way pyramidal reconstruction and effectively compute the shading equation for a small subset of pixels. For each pixel that is computed, the number of samples is adapted to the predicted variance of the integrand, which we derived from a frequency analysis of the incident illumination.

This work has been published at I3D **[CI-02]** and received a 2nd best paper award, which gave us the opportunity to publish an extended journal version at IEEE Transactions on Visualization and Computer Graphics **[RI-02]**. This extended version adds an interesting method for adaptive sampling for preconvolved shading, where our frequency prediction is used to further improve the calculation when one of the two directional functions (the illumination, or the BRDF) acts as a band limiting filter. In this case, is becomes possible to pre-filter the function with higher bandwidth with a low-pass filter and drastically reduce the number of samples involved in the shading integral.

3.4 Computational geometry

My contributions to the field of computational geometry do not constitute a coherent piece of work, because they are the result of side ideas that popped during my time of work, some being driven by my research in lighting simulation, some being a response to typical problem in the industry that was raised during my consulting activity at Digisens.

The instantiation of geometry for instance was a requirement for the simulation of radiative exchange in very large vegetation scenes. It was necessary to be able to correctly determine which parts of a big tree could be instanced with which level of approximation. My research for a smoothing algorithm for tomographic reconstruction was motivated by my consulting activities in Digisens SA, who sell a software suite for tomographic reconstruction and visualization.

This doubly proves one single thing: working on side projects might bring new interesting ideas onto which one can perform interesting research.

3.4.1 Seamless parameterization-free texturing

Texture mapping is a technique employed by graphics hardware that consists in picking up colors of an object from an image, called a *texture*. This is a very efficient way to add small surface details, without the need to explicitly define geometric primitives for each graphical element to represent on the final object. In video games for instance, meshes are usually made of large polygons, and most of the details are encoded into textures, including reflectance properties, transparency, precomputed illumination, etc.

In production, the design of textured objects is a two-fold problem: First of all, the problem of texture mapping is a problem of parameterization. Indeed, the target surface being most of the time not developable, there is no obvious parameterization with minimal deformation from the plane to the object's surface (Think for instance about putting wallpaper on a non flat surface such as a sphere). As a consequence, one generally tries to solve for a parameterization that satisfies a less drastic constraint such as being conformal, which essentially means everywhere locally isotropic, but allows some non uniform scaling across the model [Paillé 2012]. This can be achieved through a rather complex optimization process [Lévy 2002].

Another issue it that one usually starts with a texture sample, and needs to generate enough texture material for the entire surface of an object, or at least for a larger part of it, so that is can be mapped on the object without showing to much repetition. Texture synthesis from example has long been an avenue of research but few works had proposed to do it directly on the target surface at this time (See for instance [Wei 2001]).

Our contribution to texture mapping gives a possible solution to that later problem, based on the fact that if the input texture sample T is sufficiently self-similar, it should be possible to find a piecewise continuous mapping function $f : R^2 \to S$ such that $f \circ T$ is continuous. In other words, the cracks in the parameterization

are exactly compensated by the self-similarities in the input texture sample.

We achieved this using a hierarchical algorithm, by first re-meshing the input surface into a hierarchy of triangular patches, and then finding parts in the input texture samples that can be mapped onto these triangles while respecting continuity with the neighbor triangles that are already mapped. When a large patch cannot be found, the algorithm splits current patch (which reduces the constraints) and searches for smaller patches to fill in the holes. This is illustrated on Figure 3.14. What the algorithm eventually outputs is not a texture, but a parameterization, that

Figure 3.14: *Result of our texture mapping method. Parts of the input texture sample (*left*) are carefully selected and mapped onto the surface so that the result (*middle*) is eventually continuous. The image at right shows the various regions where the mapping itself is continuous. The output of the algorithm is simply a list of texture coordinates to be picked up in the input texture sample.*

can be used directly with the input texture sample for the entire surface **[RI-11]**.

3.4.2 Automatic instantiation of geometry

This work was originally inspired by the need to manage scene complexity in global illumination methods. Basically, instantiation allows to represent rotated copies of some geometry using a single pointer and a geometric transform, sparing the memory for the explicit representation of that geometry.

Instancing a scene while building it in a geometric modeler is certainly the best way to go, whenever this is possible. Most of the time however, one grabs geometric descriptions of scenes from the internet as a collection of triangles that have no particular structure. This type of data is traditionally designated by the term "polygon soup". Finding out which subsets of the input triangles can be grouped together so as to form replicated instances of similar objects is a difficult problem. We have decided to tackle it using a heuristic.

In a first step, triangles are grouped into objects based on connectivity. Then the reconstructed objects are tested for self-similarity, and hierarchically assembled into larger sets which pairwise similarity transforms are also computed. This approach is of course a heuristic, since one can easily imagine hand-crafted situations where the tessellation of a geometric scene does not allow a usable intermediary step with instantiable objects. However, this approach proved to work very nicely on most of the data that we found on the internet, simply because people build scenes by assembling together objects with a semantic meaning that is lost afterwards. Eventually,

the whole operation needs to solve the following sub-problems: (1) determining that two objects are similar up to a geometric transform, even if meshed differently and (2) determining if two collections of objects (possibly different) are globally similar up to a given transform.

We adopted a constructive approach to solve this, based on the following rule: if you know the symmetries of each piece of two different geometric models, you should be able to compute all the transforms (if any) that globally match the two models. That means we eventually need to solve the following sub-problems:

1. determine when two geometric models are similar up to a given transform;
2. determine the symmetries of an object;
3. determine the global symmetries of a group of objects knowing the symmetries and positions of each object in this group.

Straightforward solutions to the first two problems existed already: one can check for the symmetries for all possible axis orient and rotation angles, and find for which parameters the transformed object globally matches itself [Kazhdan 2004]. This has two drawbacks however: (1) it is very costly. In particular the object might have a complicated shape and very fine mesh representation, for which computing the Hausdorff distance to another rotated mesh is computationally expensive. (2) by testing possible rotations, one introduces a bias since only rotations from this set will be found. As a consequence, finding complicated symmetries potentially needs to check for a very large number of candidates, most of which will not give any result.

The solution we designed was deterministic: we first compute a set of directional functions from the shape. We call them *generalized moments*, simply because they generalize the moments of inertia of a thin-shell object to orders higher than order 2. In essence, we compute the functions:

$$\omega \mapsto \mathcal{M}^{2p}(\omega) = \int_{\mathbf{s}\in\mathcal{S}} \|\mathbf{s} \times \omega\|^{2p} d\mathbf{s}$$

The integral is computed over the entire shape $\mathcal{S}$ for each direction ω. These functions have not been chosen randomly. They have the following set of very interesting properties, which we use later to find the shape's symmetries: (1) they decompose into a finite set of spherical harmonics, and (2) any symmetry of the original shape leaves the moment functions invariant.

Spherical harmonics indeed have this nice property that rotations can be computed efficiently, and rotational symmetries of the function appear as straightforward relationships between the coefficients of the spherical harmonic decomposition [Sloan 2008]. As a consequence, we find the global symmetries of a shape using the following steps:

1. we compute the spherical harmonic coefficients of the generalized moments of a shape. For this we need a single surface integral per spherical harmonic

coefficient. This is because the powers of the sine functions involved have a known finite decomposition in the spherical harmonic basis.

2. we find the axis and angles of symmetries of the moment functions by looking at the relationship between their spherical harmonic coefficients.

3. we check that these symmetries only are valid for the original shape.

Figure 3.15 below illustrates this on a simple shape:

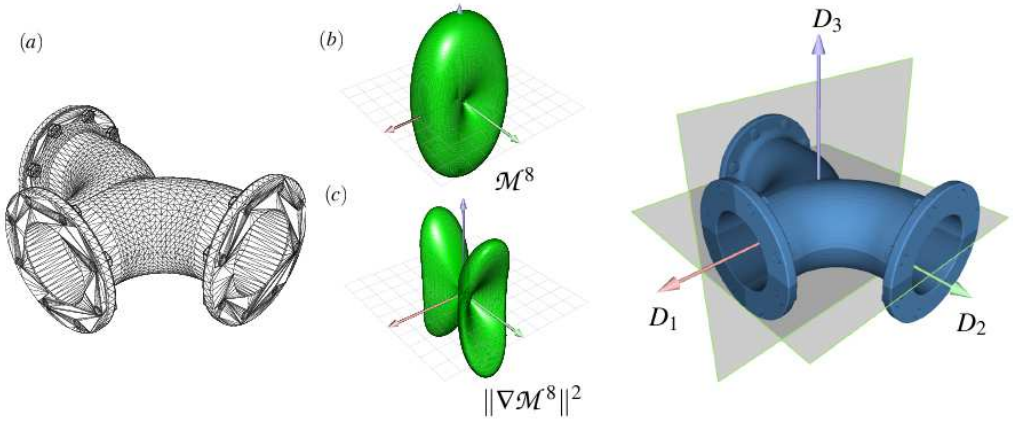


Figure 3.15: *Illustration of our method for detecting global symmetries of geometric models. A collection of directional functions are computed from the input geometry, and by studying the spherical harmonic coefficients of these function, we can deterministically discover all the axis and planes of symmetry of the object.*

In order to leverage this into finding geometric instances in a scene, we designed two algorithms: First of all, a rotation-free similarity distance can be easily obtained by summing the square of spherical harmonic coefficients of the moments for each frequency band. Indeed, spherical harmonic rotation matrices are unitary matrices. As a consequence, the sum of per-frequency band square of SH coefficients is constant when rotating a shape. Then, in a second step we designed an algorithm to determine the global symmetries of a group of objects from the symmetries and position of each of these objects. We applied this algorithm to successfully instance some huge geometric data scenes at multiple hierarchical levels, such as the *power plant model* **[RI-07]**.

This work obviously has a number of limitations. First of all, our similarity finding method is inherently global. It cannot be used to find local symmetries within a big model. However, it is possible to extend the computation of generalized moments to local regions of the surface with a Gaussian weight, for instance using:

$$\omega \mapsto \mathcal{M}^{2p}(\mathbf{x},\omega) = \int_{\mathbf{s}\in\mathcal{S}} \|\mathbf{s}\times\omega\|^{2p} e^{-\|\mathbf{x}-\mathbf{s}\|^2/\sigma^2} d\mathbf{s} \tag{3.6}$$

Although in this case the spherical harmonic decomposition of the moments will not be finite anymore, its calculation and the application of the same framework could be used to approximately detect regions with local symmetries, and approximately detecting similar regions, and eventually extract similar sub-components of the shape. To some extent, the local moments computed in Equation 3.6 would act a local geometric descriptors. But their spherical harmonic representation would help a lot determining their similarity up to a geometric transform. This is a potential avenue for future work.

3.4.3 Smoothing data for tomography reconstruction

Computer tomography is a fantastic non-destructive acquisition method. It is used in various contexts, the most popular of which is medical analysis. But it is used as well for post-production checking of products and reverse engineering in general, including acquisition and comprehension of concurrent technology. Another application is cultural heritage analysis.

Tomography reconstruction actually regroups multiple reconstruction algorithms that operate on different types of data obtained using various acquisition technologies. In all cases, the output is a volume of data (e.g. densities, X-Ray absorbance values) that represents the actual sample. X-Ray tomography for a start, consists in reconstructing a volume of data from 2D X-Ray images acquired from different directions. The problem is not exactly linear until one approximates the X-Ray absorption process by a linear equation. Eventually the problem is to reconstruct a volume from its directional 2D projections. Positron emission tomography (known as PET scan) records dual emission of particles in pairs of opposite random directions from radioactive fluor, in concentration proportional to biological activity. The problem is therefore that of reconstructing 3D data from linear projections, which is even harder. Other techniques, such as fluorescence tomography combine an indirect stimulation of the data (the sample is lit externally by UV light that diffuses through the tissue) and an indirect measurement of the effect of the data (the fluorescent light diffuses out of the sample and can then be measured), making it an even more difficult problem to solve.

Image quality issues are a common problem to all tomography reconstruction algorithms, including for the most straightforward ones such as X-Ray tomography. If only restricting to X-Ray tomography, artifacts of various natures decrease image quality: ring artifacts caused by pixel defects in the X-Ray captor, noise due to imperfect convergence and poor signal to noise ratio at the captor, beam effects due to neglecting non linear light interaction in the volume, etc. Smoothing algorithms are therefore essential to tomography.

In our work, we designed a smoothing method based on anisotropic diffusion, which consisted in convolving the data at each point with an anisotropic Gaussian 3D kernel that is computed from the data itself, inspired by existing work on image segmentation based on anti-geometric diffusion [Manay 2003].

Adapting the shape of the kernel allows to smooth the data while respecting

edges and other thin structures. We have also reformulated the diffusion process so that it is capable of running on the GPU, to make the filtering very fast, and used splitting of large volumes in order to use it on large volumetric data sets **[PI-02]**.

3.5 Computational imagery

I worked on three different project related to image analysis and manipulation, that can be useful at various stages of the image creation process in computer graphics. The three projects I'm describing here all consist in automatically understanding the content of an image, in order to be able to ease additional manipulations.

3.5.1 Analysis and synthesis of stochastic distributions of sprites

Texture synthesis by example [Efros 1999] is a well known problem that has been the subject of a lot of research in the past. The intrinsic goal is to capture the underlying process that created a input texture sample, and to use it in order to generate a new random texture that is similar in appearance, while random in the positionning of the various identifiable texture features. When the input texture sample has some structure (e.g. an irregular wall) non parametric texture synthesis methods are challenged because they cannot capture the consistency in the image at multiple scales at once, while keeping enough room for randomness, and one needs to rely on high level priors on the image structure [Liu 2004].

When a input texture is a stochastic collection of shapes (*e.g.* sprites), the human vision system does a very frustrating job in figuring out that a given image contains similar shapes otherwise randomly distributed and partly overlapping. Our contribution is this field has been to design an algorithm to automatically detect that an image contains multiple instances of similar potentially overlapping 2D objects. The actual shapes and positions of the objects are recovered and occluded parts are reconstructed based on their occurrence in multiple instances throughout the input image **[CN-01,RR-05]**. This work stands half-way between pixel-based texture synthesis [Wei 2000] and analysis/synthesis of vectorial shape arrangements [Hurtut 2009].

3.5.2 Image segmentation from inaccurate input

Computer aided foreground selection is a common operation that is offered by traditional image editing programs (e.g. Gimp, Photoshop, etc). It allows to cut and paste objects of potentially complicated geometry between images. However, separating an object from the background is an ill posted problem since the definition of the object mostly relies on semantics that cannot be robustly captured using mathematical concepts (Although it some cases, this remains possible even in very challenging situations [Shahrian 2013]). As a consequence binary object selection is usually driven by a prior coarse selection from the user, which the computer improves afterwards using an automated process (See for instance [Levin 2006, Wang 2007a]).

Although a user input greatly helps the selection process, if the input user-supplied selection is not perfect, which happens for instance when the foreground strokes partly touch the background, traditional algorithms get confused and tend to produce an incorrect output, mixing the foreground and the background together.

Our contribution has been to create a robust method that allows to produce sensible segmentation results even when the user input is imperfect. This looks contradictory of course, and obviously such a method is at fight between trusting the interpretation of the user's intention (that is the input strokes), and the algorithm's own *a priori* segmentation measure that it should rather trust if the user's input is imperfect. Figure 3.16 below gives an example of using our technique.

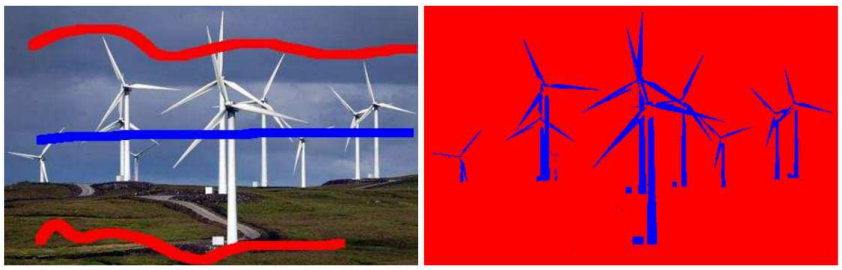


Figure 3.16: *Extreme case of selecting foreground from background using a very inaccurate input using our technique.*

We achieved this using conditional random fields to classify pixels into two categories: foreground and background. The distance we used between pixels was based on a mix between local histograms, and a Gaussian distance. Because conditional random fields require a low dimensional Euclidean distance between the feature of pixel pairs to match, we first convert all pixel descriptors at once into a low-dimensional Euclidean space using multi-dimensional scaling, based on the Landmark MDS algorithm **[I-01]**.

3.5.3 Empirical mode image decomposition

Structure-preserving image filtering is a difficult task that essentially consists in detecting which discontinuities in an image are noise and patterns and which are actual structure. Some images for instance possess noise that comes from the camera on top of fine texture details that can in turn be understood to be unwanted oscillations at a larger scale, etc. We have therefore tried to decompose images into bandwidth layers that respect the higher level structures. Such an operation offers very interesting possibilities for image editing such as texture replacement, filtering, detail enhancement, tone mapping, etc **[RI-04]**.

Edge-aware mode decomposition is however not a trivial task. In 1D, the Hilbert-Huang transform extracts modes from a signal by averaging envelops and then subtracting the mean to the original signal [Huang 2005]. What we needed was a 2D version of the Hilbert-Huang transform that would in addition be respectful of larger structures in the image.

However, no consistent 2D extension of the Hilbert transform exists, and for good reasons: there's no consistent way in 2D to define a single phase and frequency of a signal at each point. As a consequence, various extensions of the Hilbert-Huang transform exist in 2D but none do respect the larger scale structures.

Our contribution was to design such a 2D extension which is suitable for image manipulation. Our method computes two envelopes (min and max) of the image signal, using an edge-aware interpolation method proposed by Levin *et al.* [Levin 2004] for image colorization. We proposed to average them and subtract to the original image to get the details level. Figure 3.17 below shows multiple edge-aware bandwidth levels successfully extracted from the same image. This work has since been

original image 1st level smoothed/details 2nd level smoothed/details

Figure 3.17: Multiple levels of image details extracted from the snake at left. The image noise is the first, then the scales, then the shading over the snake.

cited a few times and further improved, one of its limitations being that is was hard to distinguish extrema in regions containing a mixture of texture and structure **[RI-04]**.

Chapter 4

Research project

"*Nanum gigantum humeris insidentem*"

The domain of computer graphics seems to be very rich in possibilities for interesting research. Because the demand from application domains and the hardware on which algorithms can be implemented both evolve rather rapidly, it is a real challenge to propose a long term vision of research. Of course, some very difficult questions will probably remain on the table for a long time. Among these: large scale acquisition of geometry/illumination/reflectance data, efficient processing of large data bases, real time global illumination. I consider that making even small steps in these directions is already a significant achievement.

Large scale data acquisition is probably the next challenge in computer graphics. On one hand the demand of realism in synthetic images pushes toward the handling of gigantic data sets. On the other hand, new hardware (for instance the Kinect) offers interesting possibilities to acquire them at a very low cost. The gap in between these needs to be filled by research for converting data from the captors (mostly images) into geometric models, textures, material representations, and illumination. In this field, I want to work on BRDF acquisition and procedural texture synthesis by example.

Processing large amounts of data for computer graphics will probably need to develop new mathematical tools. One example of such a challenge is efficiently filtering large scale data. In this context I am proposing to work on isotropic filter decomposition.

Although the simulation of light transport is well understood, the nature of the light transport operator is not entirely understood yet. This is the case in particular for very specular scenes. That is why I want to pursue the Fourier analysis of light transport, and also initiate new work on the dimensional analysis of the light transport operator. A better understanding of the underlying mathematical properties of light transport is likely to produce new efficient methods for global illumination.

Before to list specific research proposals, I would like to review a few principles that I have become attached to across my recent experience as a researcher, and that constitute—to me—a good basis to work with:

Experiment. Experimenting allows not only to spot new problems but also to start figuring out the possibilities of research to overcome these problems. Because

it helps identifying the "technological locks" that lie in a given field, experimentation is by all means the first step in my research.

Reformulate. Problems in computer graphics are often a mix up of multiple smaller question, that can as well be more abstract and more general and fundamental than the particular application they are extracted from. Reformulating problems helps giving them a stand-alone justification and extend them to a broader scope at the same time. This is for instance the case of transversal domains such as filtering, sampling, etc.

Import ideas from other domains Keeping an eye on research that is being developed in other domains can help a lot. It happens that interesting tools and algorithms produced in a different context have a substantial importance in the domain computer graphics. Metropolis Light Transport [Veach 1997] is such a typical example, where the Metropolis-Hastings method—a Markov chain density estimation method—for reconstructing probability densities was successfully adapted to perform photorealistic lighting simulation. I find the recent advances in sparse signal reconstruction, known as Compressive Sensing methods, really inspiring. One of my future research projects aim at adapting these techniques to perform BRDF acquisition.

Validate. In computer graphics validation is mandatory. It brings practical proof that a method can be implemented (most theories include "hidden constants" that are known to exist, but cause lots of issues when an actual number needs to be put in the code). It also helps measuring how efficient a new method is with respect to previous work, although providing fair comparisons is always difficult, since speed depends on the quality of implementation and hardware capabilities. Most conferences in computer graphics will not accept a theory paper without practical validation anyway.

Collaborate with industrial partners. Collaboration with the industry brings fresh air to research. More specifically, it brings new questions, with very specific constraints, and potential application domains for more fundamental research. My previous collaboration with video game companies (e.g. Eden Games) has been an illustration of this principle.

Collaborate with academic partners. The domain of computer graphics benefits from a spirit of sane competition. Besides, it is fairly easy to team up with researchers from "concurrent" universities to collaborate on new contributions. I've recently successfully collaborated with MIT, Cornell University and University of Montreal. As we say, "if you can't beat them, join them".

Advise students. Not only is advising PhD students part of the researcher's job, but it also puts some regularity constraints on the research and constantly needs to challenge yourself. It is also a great pleasure to see PhD students gradually turn from students to very valuable work collaborators across years.

Now I will elaborate on my future research subjects: spherical filtering, dimensional and frequency analysis of light transport, sparse BRDF reconstruction, and the analysis of procedural textures. For each of them I specify the estimated time I will devote to that particular work.

4.1 Short term research

Filtering (1-2 years)

Filtering usually consists in removing artifacts from an input signal so as to recover the ideal data. It is an ubiquitous component of computer graphics methods. Filtering is for instance necessary for displaying images at multiple resolutions while removing aliasing, to ensure consistency between several representations of data at multiple scales, to allow controlled approximations, and to facilitate measurement operations such as numerical integration. Without filtering, *aliasing* occurs, as the result of unwanted frequency content causing visual artifacts. A close look into the Fourier domain shows aliasing to be the consequence of replicated copies of the spectrum of the signal bumping into their neighbors when the sampling in the primal domain is too sparse.

A large number of image and multi-dimensional data filtering methods are available. Some are linear, such as methods based on numerical integration with a sliding filter function [Damelin 2011], some are not such as the bilateral filter [Aurich 1995]. They can be either local such as diffusion methods, or global as the total variation filters. There already exists very efficient algorithms for most of these. Even some non linear filters such as the bilateral filter can be used in real time on large data sets [Chen 2007]. Computation is especially easy when the filter is constant and the filtering operation is inherently linear. In this case, filtering is a simple product in the Fourier domain.

I am interested in particular in the problem of efficient filtering with non constant and non isotropic filters. This problem raises in computer graphics under several forms. For instance, anisotropic diffusion filters for edge-aware image and volume smoothing have a shape that depends on the data itself. That shape is an anisotropic Gaussian, usually computed as a function of the Hessian of the data, which allows to respect edges while filtering.

The same situation pops up in the spherical domain when computing the shading on an object under distant illumination, since the BRDF lobe—that stands for the filter—depends on the view direction. Because of the anisotropy of that filter, and because it is not even constant across the object, prefiltering the distant illumination with all possible shapes and orients of the filter is practically not feasible.

Finally, another similar situation is that of anti-aliasing environment mapping, because the curvature of the surface that is inside a given pixel changes the size and shape of the region of the incident illumination that needs to be averaged for that pixel.

It is therefore necessary to develop new techniques to efficiently perform non constant filtering with anisotropic filters, especially in the spherical domain. A possible solution to this problem is to first perform a decomposition of anisotropic filters into rotationally symmetric filters, so as to reduce the dimensionality of the precalculation. In a collaboration with D.Nowrouzezahrai (U. of Montreal, Canada), I have already started exploring the possibilities of using rotated zonal harmonics as

a new basis to represent anisotropic filters. In the 2D image domain (and possibly in the 3D domain if filtering videos) such a nice basis of isotropic functions does not seem to exist. In this case, relying on partial bases (bases that do not span the entire space)—for instance mixtures of Gaussians—appears to be a possible solution too.

By example procedural texture synthesis (1-2 years)

Procedural texture synthesis consists in generating textures using analytical methods possibly driven by a set of parameters [Ebert 2002] which values can be chosen at random. A procedural texture is inherently a process that can be used to endlessly generate similar textures by randomly changing the parameter values. Consequently procedural texture synthesis is used a lot in video game pipelines, since it allows to generate—possibly on the fly—a lot of similar stationary textures with sufficient variability.

Existing softwares such as the tool provided by Allegorithmics [1] provide a system to build a pipeline of image operators which eventually produce a stationary texture. Changing the random seed between two realizations of the pipeline produces different textures that still seem to be produced by the same underlying process.

However, given an input texture sample, it is extremely hard to figure out what is the set of image operations that eventually produced that particular image. This problem is known as *procedural texture synthesis by example*. In game designing pipelines, finding a procedural texture that mimics a given input usually relies on the capability of artists which deep knowledge of the texture generation software helps finding the correct blocks to tie together. Doing the same thing in a completely automated process is a very challenging problem.

The problem of procedural texture synthesis by example has been addressed for very specific cases only, in particular Perlin noise [Lagae 2009] and Gabor noise [Lagae 2012], but no generic solution exists. The difficulty behind this comes from the fact that most of the operations that take place in a procedural pipeline are intrinsically non linear (examples include color map combination, thresholding, etc) and sometimes even modify the image in a way that might appear as random afterwards, possibly removing the information that was needed to generate a particular effect.

One option to recover the process that created a given texture is of course to perform a brute force search over all possible image combination methods, the same way a "Plouffe" inverse symbolic calculator proceeds [2] in order to find mathematical expressions that match a given decimal number.

However, a given instance of a procedural texture is often based on random values of the parameters, which would also need to be discovered. That means in particular that even when generating random textures, only the underlying process

[1] `http://www.allegorithmic.com/`

[2] See for instance http://www.mrob.com/pub/ries/index.html

should be matched, independently of the random parameters that have been used to instance it.

In order to overcome this problem, one needs an image similarity measure that can determine whether or not two images are the result of the same image creation process, possibly using a different set of random parameters. I propose to look for such an image descriptor, and to use it in order to generate procedural descriptions of textures using an inverse symbolic calculator.

Among possible solutions, we could use multi-level covariance distributions [Karacan 2013b]. We might for instance compute a Gaussian/Laplacian pyramid of the image, and then compute clouds of covariance matrices on each level. The descriptor will be the multi-level cloud of values. In a first step, several experiments are needed to find out whether such a descriptor would have enough discrimination power. In any case, this is an ambitious project, that includes potentially interesting applications.

Perceptual Analysis of BRDF Approximations (1 year)

There exist a large number of analytical material reflectance models: Ashikmin-Shirley, He, Micro-facet models, for which the L2 approximation error has been well studied [Ngan 2005]. However, the ability a person has to perceive a material is not well described by the L2 error (see for instance Fig. 3.13), and ultimately, the decision to use a particular approximation in realistic rendering should be driven by how realistic the result will look like.

We denote by "realism" the amount at which an object is recognized by a human viewer. A study is therefore necessary to sort out the following questions: To which extend does realism depend on the approximations to the material model rather than the geometry or the illumination ? What is the impact of the various BRDF approximations on the perceived realism of a rendered image?

That needs rendering multiple BRDFs in multiple lighting conditions on different geometry, and setting up a set of perceptual experiments. This is a small project, but it could be a nice opportunity for a collaboration with people in human perception and human vision teams, with a potential impact on video games, and virtual prototyping.

4.2 Long term research

Acquisition and representation of materials (3-4 years)

Numerical models of materials are the key ingredient to realism in image synthesis, whatever the rendering technique involved, should they work in real time for video games, or offline in a computationally expensive but unbiased physically-based lighting simulation. Numerical models for materials are designated by the term "reflectance" which groups multiple sub-families of functions with one to height parameters.

Acquisition and processing of reflectance data from real-life material samples is practically difficult for several reasons: (1) the acquisition cost is huge. Several hours are necessary to acquire an accurate BTF (Bidirectional Texture Function) with a gonioreflectometer [Matusik 2003]; (2) the acquisition requires calibration both in the photometric and geometric domains; (3) acquired data must be adequately filtered, parameterized and compressed in order to be used later in a lighting simulation system. As a consequence, the acquisition of material properties requires a complex setup and lots of processing power. It seems necessary to derive new acquisition algorithms that can fit consumer hardware (a camera, a webcam,...) and that run at a low computation cost, while controlling the possible approximation errors.

A technique that would ideally be able to run on low price hardware would need a complete pipeline that is able to recover all the missing information that is usually provided by a fully calibrated system: high dynamic range reconstruction of the signal, calculation of surface normals and reflectance properties, and possibly calculation of the incident illumination as well, although the later is easy to capture with good accuracy using a spherical reflector [Ren 2011].

I want to explore the application of the theory of *compressive sensing* to acquire reflectance functions. Compressive sensing is a set of techniques which aim at reconstructing sparse signals from linear measurements. A sparse signal is a signal which expresses with a small number of significant coefficients in a linear function basis. Practical experiments have shown that such signals can be reconstructed with a number of measurements that is well below the traditional Nyquist limit. The reason for this is that the inherent dimensionality of sparse signals is small in comparison to the space they live in. The good news is that most signals we're dealing with in computer graphics are sparse. This is the case for reflectance functions in particular, but also light fields [Marwah 2013] and solutions of global illumination problems [Peers 2009].

This work should start with a preliminary experimental phase where we should test the feasibility of compressive sensing reconstruction in simple situations. We will measure for various choices of function spaces and various types of data measurement how nicely existing compressive sensing methods work. Available algorithms include for instance the SpaRSA method [Wright 2009], or the COSAMP method [Needell 2010]. Then, we will explore how feasible it is to extend the recovery to not only the reflectance but also the normals of the surface. That will need to apply non linear compressive sensing techniques [Blumensath 2012], which have been proved to work under reasonable assumptions. Finally, this work should eventually provide a complete acquisition pipeline and the software suite that allows to perform acquisition of reflectance properties.

This work has already begun, and I am currently advising a PhD student on that particular subject. First results have already given birth to a poster at Siggraph Asia'2013.

Frequency and dimensional analysis of the light transport operator (3-4 years)

Lighting simulation always results into computing more than what is eventually necessary. This crude statement is actually motivated by the fact that the amount of information in synthetic images, in the sense used by information theory, is usually much less than what the space of functions over a grid of pixels actually contains. In other words, simulated images are sparse, with very limited frequency content. This is even worse when the dimensionality of the problem increases, as we have proved in our works on generating depth of field and motion blur images. Sub-surface scattering and light scattering in general also tend to produce synthetic images with rather smooth variations.

The apparent regularity of synthetic images is the consequence of two distinct phenomena. First of all, light transport mostly acts as a low-pass filter over the illumination, with the notable exception of occlusion. Secondly, the light transport operator has a rather low dimensionality, which means that is can be represented compactly in a well chosen function basis. That has been successfully used for compressing precomputed radiance transport [Sloan 2002]. Traditional computation methods, and in particular methods based on Monte-Carlo summation in the space of light paths do not account for these characteristics and for good reasons: it is difficult to know in advance that the combined effect of many light paths is going to produce a smooth function in the image space, and how smooth that function will eventually be. The frequency analysis of light transport aims at giving such tools.

Frequency analysis of light transport Frequency analysis of light transport consists in reformulating local light transport operators in the Fourier domain, so as to characterize the spectrum of a light distribution before the light distribution itself is even computed. This analysis allows to allocate adequate computational power to the calculation using adaptive sampling and proper reconstruction filters. It is usually performed in 4D but it elegantly extends to 5D including the temporal dimension so as to predict the variations of light in images taken with a finite aperture time.

In my own contributions, I have already explored theoretical and practical aspects of frequency analysis of light transport, in particular related to depth of field and motion blur effects. We have also derived various representations for the frequency content of local light fields some suitable to real time rendering, some suitable for unbiased offline rendering. This works has very recently been extended to participating media **[RI-00]**. In the near future, I would like to try to apply frequency analysis to sub-surface scattering. A more generic research topic is also to use frequency analysis along light paths to drive Markov chains Monte-Carlo methods such as Metropolis Light Transport [Veach 1997] and Energy Redistribution Path Tracing [Cline 2005]. Indeed, mutation strategies are usually driven by how much energy is conveyed by a light path. It is likely that a faster converging solution can be obtain by oversampling regions with a higher variance instead.

Dimensional analysis of the light transport operator The light transport operator, as defined in Section 3.1.1 transforms a full distribution of light having bounced n times into a full distribution of light having bounced $n + 1$ times. With this particular definition, the light transport operator appears to be dense since it spreads light everywhere in a scene.

The idea behind the dimensional analysis of light transport is to find function spaces in which it is as sparse as possible. In other words, if we can find such a function space where one transported distribution of light decomposes over a small number of basis functions, we might be able to increase computation speed a lot. One good candidate of course is the eigen space of the light transport operator. I propose to explore that particular space. It is also likely that examining local transport operators such as the reflectance operator or the scattering operator will bring interesting results.

These eigenspaces that inherently depend on the scene's geometry and materials but not on the light sources, will potentially give new representations for the light transport operator in which it is more compact and therefore faster to compute. However, Fredholm operators with a discontinuous kernel (that is what we're dealing with) do not usually have a spectrum that is easy to compute. What do the eigen functions of light transport in a geometric scene look like? I have no idea yet. But I have the feeling that some interesting information is to be found there.

Appendix A

Publications

Nota bene: the Siggraph conference publishes its proceedings as a special issue of the journal ACM Transactions on Graphics. It is common to consider Siggraph publications among international journals with reviewing committee. Since 2009, authors of papers at ACM TOG are also offered the possibility to present their work at Siggraph.

A.1 Internationales journals with reviewing commitee

RI-00 A Local Frequency Analysis of Scattering and Absorption; Laurent Belcour, Kavita Bala, Cyril Soler; *ACM Transactions on Graphics, ACM Press, 2014 (to appear). Will be presented at Siggraph'2014*

RI-01 5D Covariance Tracing for Efficient Defocus and Motion Blur; Laurent Belcour, Cyril Soler, Kartic Subr, Nicolas Holzschuch, Frédo Durand; *ACM Transactions on Graphics, ACM Press, 2013, 32 (3). Presented at Siggraph'2013*

RI-02 Interactive Rendering of Acquired Materials on Dynamic Geometry Using Frequency Analysis; Mahdi Bagher, Cyril Soler, Kartic Subr, Laurent Belcour, Nicolas Holzschuch; *IEEE Transactions on Visualization and Computer Graphics, IEEE, 2013, 19 (5), pp. 749-761*

RI-03 Accurate fitting of measured reflectances using a Shifted Gamma micro-facet distribution; Mahdi Bagher, Cyril Soler, Nicolas Holzschuch; *Computer Graphics Forum, Blackwell Publishing, 2012, 31 (4)*

RI-04 Edge-Preserving Multi-scale Image Decomposition based on Local Extrema; Kartic Subr; Cyril Soler; Fredo Durand; *ACM Transactions on Graphics, ACM, 2009, 28 (5), pp. 147:1-147:9. Presented at Siggraph Asia'2009*

RI-05 Fourier Depth of Field; Cyril Soler; Kartic Subr; Frédo Durand; Nicolas Holzschuch; François X. Sillion; *ACM Transactions on Graphics, ACM Press, 2009, 28 (2), pp. 18:1-18:12. Presented at Siggraph 2009*

RI-06 Third Radiation Transfer Model Intercomparison (RAMI) exercise: Documenting progress in canopy reflectance models; J.-L. Widlowski; M. Taberner; B. Pinty; V. Bruniquel-Pinel; M. Disney; R. Fernandes; Jean-Philippe Gastellu-Etchegorry; N. Gobron; A. Kuusk; T. Lavergne; S. Leblanc; P. Lewis; E. Martin; M. Mottus; P.R.J. North; W. Qin; M. Robustelli; N. Rochdi; R. Ruiloba; Cyril Soler; R. Thompson; W. Verhoef; M.M. Verstraete; D. Xie; *Journal of Geophysical Research, 2007, 112, pp. D09111*

RI-07 Accurate Detection of Symmetries in 3D Shapes; Aurélien Martinet; Cyril Soler; Nicolas Holzschuch; François X. Sillion; *ACM Transactions on Graphics, ACM, 2006, 25 (2), pp. 439 - 464*

RI-08 A Frequency Analysis of Light Transport; Frédo Durand; Nicolas Holzschuch; Cyril Soler; Eric Chan; François X. Sillion; *ACM Transactions on Graphics, ACM, 2005, 24 (3), pp. 1115 - 1126. Presented at Siggraph'2005*

RI-09 Graphics Gems Revisited; Stéphane Guy; Cyril Soler; *ACM Transactions on Graphics, ACM, 2004. Presented at Siggraph'2004*

RI-10 An Efficient Instantiation Algorithm for Simulating Radiant Energy Transfer in Plant Models; Cyril Soler; François X. Sillion; Frédéric Blaise; Philippe De Reffye; *ACM Transactions on Graphics, ACM, 2003, 22 (2), pp. 204 - 233*

RI-11 Hierarchical Pattern Mapping; Cyril Soler; Marie-Paule Cani; Alexis Angelidis; *SIGGRAPH, Jul 2002, San Antonio, Texas, United States.*

RI-12 Texture-Based Visibility for Efficient Lighting Simulation; Cyril Soler; François X. Sillion; *ACM Transactions on Graphics, ACM, 2000, 19 (4)*

RI-13 Fast Calculation of Soft Shadow Textures Using Convolution; Cyril Soler; François X. Sillion; *Computer Graphics Proceedings, 1998, Orlando, Floride, United States. pp. 321-332*

A.2 International conferences with reviewing commitee

CI-01 Accurate Binary Image Selection from Inaccurate User Input; Kartic Subr, Sylvain Paris, Cyril Soler, Jan Kautz; *Computer Graphics Forum, John Wiley & Sons, Inc., 2013, 32 (2)*

CI-02 Interactive rendering of acquired materials on dynamic geometry using bandwidth prediction; Mahdi Bagher, Cyril Soler, Kartic Subr, Laurent Belcour, Nicolas Holzschuch; *Michael Garland and Rui Wang. I3D - ACM Siggraph Symposium on Interactive 3D Graphics and Games, Mar 2012, Costa Mesa, United States. ACM, pp. 127-134*

CI-03 A Deferred Shading Pipeline for Real-Time Indirect Illumination; Cyril Soler; Olivier Hoel; Frank Rochet; *ACM SIGGRAPH 2010 Talks, Jul 2010, Los Angeles, CA, United States. ACM, pp. 18*

CI-04 Hierarchical Instantiation for Radiosity; Cyril Soler; François X. Sillion; *B. Peroche and H. Rushmeier. Rendering Techniques '00, 2000, Brno, Czech Republic. Springer Wien, pp. 173-184*

CI-05 Automatic Calculation of Soft Shadow Textures for Fast, High Quality Radiosity; Cyril Soler; François X. Sillion; *George Drettakis and Nelson Max. Eurographics Rendering Workshop 1998, 1998, New York City, NY, United States. Springer Wein, pp. 199-210*

CI-06 Accurate Error Bounds for Multi-Resolution Visibility; Cyril Soler; François X. Sillion; *X. Pueyo and P. Shröder. Proceedings of 7th Eurographics Workshop on Rendering (Rendering Techniques '96), 1996, Porto, Portugal . Springer Verlag, pp. 133-143*

CI-07 A Clustering Algorithm for Radiance Calculation In General Environments; François X. Sillion; George Drettakis; Cyril Soler; *Eurographics Workshop on Rendering Techniques, 1995, Dublin, Ireland.*

A.3 Posters in international conferences

PI-01 Frequency-Based Kernel Estimation for Progressive Photon Mapping; Laurent Belcour, Cyril Soler; *ACM SIGGRAPH Conference and Exhibition on Computer Graphics and Interactive Techniques in Asia , Dec 2011, Hong Kong, China. ACM, SA '11 SIGGRAPH Asia 2011 Posters, pp. Article No. 47*

PI-02 Multiscale Feature-Preserving Smoothing of Tomographic Data; Nassim Jibai, Cyril Soler, Kartic Subr, Nicolas Holzschuch; *ACM SIGGRAPH 2011 Posters, Aug 2011, Vancouver, Canada. SIGGRAPH '11 ACM SIGGRAPH 2011 Posters, pp. Article No. 63*

PI-03 Screen-Space Percentage-Closer Soft Shadows; Mahdi Bagher, Jan Kautz, Nicolas Holzschuch, Cyril Soler; *ACM SIGGRAPH 2010 Posters, Jul 2010, Los Angeles, CA, United States. ACM, pp. 133*

PI-04 Accurately Detecting Symmetries of 3D Shapes; Aurélien Martinet; Cyril Soler; Nicolas Holzschuch; François X. Sillion; *SGP 2005 - Eurographics Symposium on Geometry Processing (Poster session), Jul 2005, Vienna, Austria*

PI-05 Sparse Approximations of BRDFs; Benoit Zupancic; Cyril Soler; *Siggraph Asia 2013 (Poster session), Nov 2013, Hong Kong*

A.4 French journals

RN-01 HYEMALIS: Un Simulateur d'Images de Paysages Tridimensionnels Complexes; Jérôme Helbert, Béatrice Berthelot, Cyril Soler; *Revue Francaise de Photogrammetrie et de Télé-détection, 2003, Marne-la-Vallée, France. pp. 27-35*

RN-02 Caractérisation Multi-échelles de la Visibilité pour les Calculs de Radiosité; Cyril Soler; François X. Sillion; *Revue Internationale de CFAO et d'informatique graphique, Elsevier, 1996*

A.5 Communications in French conferences

CN-1 Analyse et synthèse de textures composées de motifs répétitifs; Pierre-Edouard Landes; Cyril Soler; *AFIG'08 - 21èmes journées de l'Association Française d'Informatique Graphique (AFIG), Nov 2008, Toulouse, France. IRIT Press*

CN-02 Organisation Automatique de Scènes 3D; Aurélien Martinet; Cyril Soler; Nicolas Holzschuch; François X. Sillion; *17èmes journées de l' Association Française d'Informatique Graphique (AFIG), Nov 2004, Poitiers, France.*

CN-04 Real-Time Screen-Space Indirect Illumination for Video Games; Cyril Soler, Olivier Hoel; *Future Game On, Paris, Sept. 2010*

CN-03 Caractérisation multi-échelles de la visibilité pour les calculs de radiosité; Cyril Soler; *Troisièmes journées de l'Association Française d'Informatique Graphique (AFIG '95), 1995, Marseille, France.*

CN-05 Real-Time Screen-Space Indirect Illumination for Video Games; Cyril Soler; *CARI'2000. Antananarivo, Octobre 2000*

A.6 Research reports

RR-01 Accurate and Efficient Filtering using Isotropic Filter Decomposition; Cyril Soler, Mahdi Bagher, Derek Nowrouzezahrai; *INRIA Research Report, 2013, RR-8349*

RR-02 Fast multi-resolution shading of acquired reflectance using bandwidth prediction; Mahdi Bagher url; Cyril Soler; Kartic Subr; Laurent Belcour; Nicolas Holzschuch; *INRIA Research Report, 2011, RR-7839*

RR-03 Automatic Pen-and-Ink Illustration of Tone, Gloss, And Texture; Kaleigh Smith; Cyril Soler; Thomas Luft; Oliver Deussen; Joëlle Thollot; *INRIA Research Report, 2010, pp. 21. RR-7194*

RR-04 Hierarchical Screen Space Indirect Illumination For Video Games; Cyril Soler; Olivier Hoel; Franck Rochet; Frederic Jay; Nicolas Holzschuch; *INRIA Research Report, 2009, pp. 22. RR-7162*

RR-05 Content-Aware Texture Synthesis; Pierre-Edouard Landes; Cyril Soler; *INRIA Research Report, 2009, pp. 20. RR-6959*

RR-06 Accurately Detecting Symmetries of 3D Shapes; Aurélien Martinet; Cyril Soler; Nicolas Holzschuch; François X. Sillion; *INRIA Research Report, 2005, pp. 30. RR-5692*

RR-07 A physiological Plant Growth Simulation Engine Based on Accurate Radiant Energy Transfer; Cyril Soler; François X. Sillion; Frédéric Blaise; Philippe De Reffye; *INRIA Research Report, 2001. RR-4116*

A.7 PhD Thesis

TH-01 Représentations hiérarchiques de la visibilité pour le contrôle de l'erreur en simulation de l'éclairage; Cyril Soler; *Université Joseph-Fourier - Grenoble I, Dec. 1998*

Bibliography

[Ashdown 2001] Ian Ashdown. Eigenvector radiosity. Master's thesis, Department of Computer Science, University of British Columbia, Vancouver, British Columbia, April 2001. (Cité en page 24.)

[Ashikhmin 2001] Michael Ashikhmin, Simon Premoze, Peter Shirley et Brian Smits. *A Variance Analysis of the Metropolis Light Transport Algorithm.* The Visual Computer, vol. 25, no. 2, pages 287–294, April 2001. (Cité en page 27.)

[Aurich 1995] Volker Aurich et Jörg Weule. *Non-Linear Gaussian Filters Performing Edge Preserving Diffusion.* In Mustererkennung 1995, 17. DAGM-Symposium, pages 538–545, London, UK, UK, 1995. Springer-Verlag. (Cité en page 59.)

[Avidan 2007] Shai Avidan et Ariel Shamir. *Seam Carving for Content-aware Image Resizing.* ACM Trans. Graph., vol. 26, no. 3, Juillet 2007. (Cité en page 5.)

[Baranoski 1997] Gladimir V. G. Baranoski, Randall Bramley et Jon G. Rokne. *Eigen-Analysis for Radiosity Systems.* In Proceedings of the Sixth International Conference on Computational Graphics and Visualization Techniques (Compugraphics '97), pages 193–201, Vilamoura, Algarve, Portugal, December 1997. (Cité en page 24.)

[Been 2008] Ken Been, Martin Nöllenburg, Sheung-Hung Poon et Alexander Wolff. *Optimizing Active Ranges for Consistent Dynamic Map Labeling.* In Proceedings of the Twenty-fourth Annual Symposium on Computational Geometry, SCG '08, pages 10–19, New York, NY, USA, 2008. ACM. (Cité en page 5.)

[Bénard 2011] Pierre Bénard, Adrien Bousseau et Joëlle Thollot. *State-of-the-Art Report on Temporal Coherence for Stylized Animations.* Computer Graphics Forum, vol. 30, no. 8, pages 2367–2386, Décembre 2011. (Cité en page 7.)

[Blumensath 2012] Thomas Blumensath. *Compressed Sensing with Nonlinear Observations and Related Nonlinear Optimisation Problems.* CoRR, vol. abs/1205.1650, 2012. (Cité en page 62.)

[Botsch 2010] Mario Botsch, Leif Kobbelt, Mark Pauly, Pierre Alliez et Br uno Levy. Polygon mesh processing. AK Peters, 2010. (Cité en page 5.)

[Bousseau 2009] Adrien Bousseau, Sylvain Paris et Frédo Durand. *User-assisted Intrinsic Images.* In ACM SIGGRAPH Asia 2009 Papers, SIGGRAPH Asia '09, pages 130:1–130:10, New York, NY, USA, 2009. ACM. (Cité en page 5.)

[Bruneton 2012] Eric Bruneton et Fabrice Neyret. *Real-time Realistic Rendering and Lighting of Forests*. Comp. Graph. Forum, vol. 31, no. 2pt1, pages 373–382, Mai 2012. (Cité en page 6.)

[Carroll 2011] Robert Carroll, Ravi Ramamoorthi et Maneesh Agrawala. *Illumination Decomposition for Material Recoloring with Consistent Interreflections*. ACM Trans. Graph., vol. 30, no. 4, pages 43:1–43:10, Juillet 2011. (Cité en page 5.)

[Chen 2007] Jiawen Chen, Sylvain Paris et Frédo Durand. *Real-time Edge-aware Image Processing with the Bilateral Grid*. In ACM SIGGRAPH 2007 Papers, SIGGRAPH '07, New York, NY, USA, 2007. ACM. (Cité en page 59.)

[Chen 2013] Tao Chen, Zhe Zhu, Ariel Shamir, Shi-Min Hu et Daniel Cohen-Or. *3Sweepp: Extracting Editable Objects from a Single Photo*. ACM Trans. Graph., vol. 32, no. 6, pages 195:1–195:10, Novembre 2013. (Cité en page 5.)

[Christensen 2012] Per H. Christensen, George Harker, Jonathan Shade, Brenden Schubert et Dana Batali. *Multiresolution Radiosity Caching for Global Illumination in Movies*. In ACM SIGGRAPH 2012 Talks, SIGGRAPH '12, pages 47:1–47:1, New York, NY, USA, 2012. ACM. (Cité en page 4.)

[Cline 2005] David Cline, Justin Talbot et Parris Egbert. *Energy Redistribution Path Tracing*. ACM Trans. Graph., vol. 24, no. 3, pages 1186–1195, Juillet 2005. (Cité en page 63.)

[Correa 2007] Wagner T. Correa, James T. Klosowski, Chrisopher J. Morris et Thomas M. Jackmann. *SPVN: A New Application Framework for Interactive Visualization of Large Datasets*. In ACM SIGGRAPH 2007 Courses, SIGGRAPH '07, New York, NY, USA, 2007. ACM. (Cité en page 5.)

[Cournède 2009] Paul-Henry Cournède, Thomas Guyard, Benoît Bayol, Sébastien Griffon, François de Coligny, Philippe Borianne, Marc Jaeger et Philippe de Reffye. *A Forest Growth Simulator Based on Functional-Structural Modelling of Individual Trees*. In Proceedings of Plant growth Modeling, and their Applications (PMA09), pages 34–41, Beijing, China, November 2009. Li, B. and Jaeger, M. and Guo, Y. (Eds). (Cité en page 30.)

[Damelin 2011] S. Damelin et W. Miller. The mathematics of signal processing. Cambridge University Press, 2011. (Cité en page 59.)

[Dutré 2003] Philip Dutré, Philippe Bekaert et Kavita Bala. Advanced global illumination. AK Peters Limited, 2003. (Cité en page 26.)

[Ebert 2002] David S. Ebert, F. Kenton Musgrave, Darwyn Peachey, Ken Perlin et Steven Worley. Texturing and modeling: A procedural approach. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd édition, 2002. (Cité en page 60.)

[Efros 1999] Alexei A. Efros et Thomas K. Leung. *Texture Synthesis by Non-Parametric Sampling.* In Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2, ICCV '99, pages 1033–, Washington, DC, USA, 1999. IEEE Computer Society. (Cité en pages 6 et 54.)

[Eisemann 2006] Elmar Eisemann et Xavier Décoret. *Fast Scene Voxelization and Applications.* In Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games, I3D '06, pages 71–78, New York, NY, USA, 2006. ACM. (Cité en page 45.)

[Eisemann 2007] Elmar Eisemann et Xavier Décoret. *Visibility Sampling on GPU and Applications.* Computer Graphics Forum, vol. 26, no. 3, Septembre 2007. (Cité en page 11.)

[Eitz 2012] Mathias Eitz, Ronald Richter, Tamy Boubekeur, Kristian Hildebrand et Marc Alexa. *Sketch-Based Shape Retrieval.* ACM Transactions on Graphics (Proc. SIGGRAPH), vol. 31, no. 4, pages 31:1–31:10, 2012. (Cité en page 6.)

[Farbman 2009] Zeev Farbman, Gil Hoffer, Yaron Lipman, Daniel Cohen-Or et Dani Lischinski. *Coordinates for Instant Image Cloning.* ACM Trans. Graph., vol. 28, no. 3, pages 67:1–67:9, Juillet 2009. (Cité en page 5.)

[Foley 2005] Tim Foley et Jeremy Sugerman. *KD-tree Acceleration Structures for a GPU Raytracer.* In Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware, HWWS '05, pages 15–22, New York, NY, USA, 2005. ACM. (Cité en page 29.)

[Forrest 2003] A. R. Forrest. *Future Trends in Computer Graphics: How Much is Enough?* J. Comput. Sci. Technol., vol. 18, no. 5, pages 531–537, Septembre 2003. (Cité en page 10.)

[Génevaux 2013] Jean-David Génevaux, Éric Galin, Eric Guérin, Adrien Peytavie et Bedřich Beneš. *Terrain Generation Using Procedural Models Based on Hydrology.* ACM Trans. Graph., vol. 32, no. 4, pages 143:1–143:13, Juillet 2013. (Cité en page 6.)

[Hachisuka 2008] Toshiya Hachisuka, Wojciech Jarosz, Richard Peter Weistroffer, Kevin Dale, Greg Humphreys, Matthias Zwicker et Henrik Wann Jensen. *Multidimensional Adaptive Sampling and Reconstruction for Ray Tracing.* ACM Trans. Graph., vol. 27, no. 3, pages 33:1–33:10, Août 2008. (Cité en page 27.)

[Hachisuka 2009] Toshiya Hachisuka et Henrik Wann Jensen. *Stochastic Progressive Photon Mapping.* ACM Transactions on Graphics, vol. 28, no. 5, 2009. (Cité en pages 26 et 38.)

[Hapala 2013] Michal Hapala, Tomáš Davidovič, Ingo Wald, Vlastimil Havran et Philipp Slusallek. *Efficient Stack-less BVH Traversal for Ray Tracing.* In

Proceedings of the 27th Spring Conference on Computer Graphics, SCCG '11, pages 7–12, New York, NY, USA, 2013. ACM. (Cité en page 29.)

[Havran 2001] Vlastimil Havran. *Heuristic Ray Shooting Algorithms*. PhD thesis, Czech Technical University, Praha, Czech Republic, April 2001. Available from http://www.cgg.cvut.cz/ havran/phdthesis.html. (Cité en page 29.)

[Hays 2007] James Hays et Alexei A Efros. *Scene Completion Using Millions of Photographs*. ACM Transactions on Graphics (SIGGRAPH 2007), vol. 26, no. 3, 2007. (Cité en page 5.)

[Hergel 2014] Jean Hergel et Sylvain Lefebvre. *Clean color: Improving multi-filament 3D prints*. In Computer Graphics Forum, Proceedings of Eurographics Conference, 2014. to appear. (Cité en page 7.)

[Hertzmann 2000] Aaron Hertzmann et Ken Perlin. *Painterly Rendering for Video and Interaction*. In Proceedings of the 1st International Symposium on Non-photorealistic Animation and Rendering, NPAR '00, pages 7–12, New York, NY, USA, 2000. ACM. (Cité en page 6.)

[Hoberock 2010] Jared Hoberock et John C. Hart. *Arbitrary Importance Functions for Metropolis Light Transport*. Computer Graphics Journal, vol. 29, no. 6, pages 1993–2003, 2010. (Cité en page 27.)

[Huang 2005] Norden Huang et Samuel Shen. Hilbert-huang transform and its applications. Interdisciplinary Mathematical Sciences 5. Hackensack, NJ: World Scientific. xii, 311 p., 2005. (Cité en page 55.)

[Hurtut 2009] T. Hurtut, P.-E. Landes, J. Thollot, Y. Gousseau, R. Drouillhet et J.-F. Coeurjolly. *Appearance-guided Synthesis of Element Arrangements by Example*. In Proceedings of the 7th International Symposium on Non-Photorealistic Animation and Rendering, NPAR '09, pages 51–60, New York, NY, USA, 2009. ACM. (Cité en page 54.)

[Ishimaru 1997] A Ishimaru. Wave propagation and scattering in random media. IEEE Press and Oxford University Press, 1997. (Cité en page 38.)

[Jarosz 2008] Wojciech Jarosz, Craig Donner, Matthias Zwicker et Henrik Wann Jensen. *Radiance Caching for Participating Media*. ACM Transactions on Graphics, vol. 27, no. 1, pages 7.1–7.11, March 2008. (Cité en page 27.)

[Jensen 1996] Henrik Wann Jensen. *Global Illumination Using Photon Maps*. In Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering), pages 21–30. Springer-Verlag/Wien, 1996. (Cité en pages 26 et 38.)

[Kajiya 1986] James T. Kajiya. *The Rendering Equation*. In Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques,

SIGGRAPH '86, pages 143–150, New York, NY, USA, 1986. ACM. (Cité en pages 23 et 26.)

[Kaplanyan 2010] Anton Kaplanyan et Carsten Dachsbacher. *Cascaded Light Propagation Volumes for Real-time Indirect Illumination*. In Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, I3D '10, pages 99–107, New York, NY, USA, 2010. ACM. (Cité en page 6.)

[Karacan 2013a] Levent Karacan, Erkut Erdem et Aykut Erdem. *Structure-preserving Image Smoothing via Region Covariances*. ACM Trans. Graph., vol. 32, no. 6, pages 176:1–176:11, Novembre 2013. (Cité en page 5.)

[Karacan 2013b] Levent Karacan, Erkut Erdem et Aykut Erdem. *Structure-preserving Image Smoothing via Region Covariances*. ACM Trans. Graph., vol. 32, no. 6, pages 176:1–176:11, Novembre 2013. (Cité en page 61.)

[Kautz 2002] Jan Kautz, Peter-Pike Sloan et John Snyder. *Fast, Arbitrary BRDF Shading for Low-frequency Lighting Using Spherical Harmonics*. In Proceedings of the 13th Eurographics Workshop on Rendering, EGRW '02, pages 291–296, Aire-la-Ville, Switzerland, Switzerland, 2002. Eurographics Association. (Cité en page 19.)

[Kazhdan 2004] Michael Kazhdan, Thomas Funkhouser et Szymon Rusinkiewicz. *Symmetry Descriptors and 3D Shape Matching*. In Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, SGP '04, pages 115–123, New York, NY, USA, 2004. ACM. (Cité en page 50.)

[Kelemen 2002] Csaba Kelemen, Laszlo Szirmay-Kalos, Gyorgy Antal et Ferenc Csonka. *A Simple and Robust Mutation Strategy for the Metropolis Light Transport Algorithm*. Computer Graphics Forum (Proceedings of Eurographics 2002), vol. 21, no. 3, September 2002. (Cité en page 27.)

[Keller 1997] Alexander Keller. *Instant Radiosity*. In Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '97, pages 49–56, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co. (Cité en page 27.)

[Kirk 1994] David B. Kirk et James R. Arvo. *Unbiased Variance Reduction for Global Illumination*. In P. Brunet et F. W. Jansen, editeurs, Photorealistic Rendering in Computer Graphics (Proceedings of the Second Eurographics Workshop on Rendering), pages 45–53. Springer-Verlag, 1994. (Cité en page 27.)

[Lafortune 1993] Eric P. Lafortune et Yves D. Willems. *Bi-directional Path Tracing*. In H. P. Santo, editeur, Proceedings of Third International Conference on Computational Graphics and Visualization Techniques (Compugraphics '93), pages 145–153, Alvor, Portugal, December 1993. (Cité en page 26.)

[Lagae 2009] Ares Lagae, Peter Vangorp, Toon Lenaerts et Philip Dutré. *Isotropic Stochastic Procedural Textures by Example*. Report CW 546, Department of Computer Science, K.U.Leuven, Celestijnenlaan 200A, 3001 Heverlee, Belgium, May 2009. (Cité en page 60.)

[Lagae 2012] Ares Lagae, Bruno Galerne, Sylvain Lefebvre et George Drettakis. *Gabor Noise by Example*. ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2012), page 9, 2012. (Cité en page 60.)

[Lehtinen 2011] Jaakko Lehtinen, Timo Aila, Jiawen Chen, Samuli Laine et Frédo Durand. *Temporal Light Field Reconstruction for Rendering Distribution Effects*. In ACM SIGGRAPH 2011 Papers, SIGGRAPH '11, pages 55:1–55:12, New York, NY, USA, 2011. ACM. (Cité en page 27.)

[Lehtinen 2013a] Jaakko Lehtinen, Tero Karras, Samuli Laine, Miika Aittala, Fredo Durand et Timo Aila. *Gradient-Domain Metropolis Light Transport*. ACM Transactions on Graphics, vol. 32, no. 4, pages 95:1–95:12, 2013. (Cité en page 27.)

[Lehtinen 2013b] Jaakko Lehtinen, Tero Karras, Samuli Laine, Miika Aittala, Frédo Durand et Timo Aila. *Gradient-domain Metropolis Light Transport*. ACM Trans. Graph., vol. 32, no. 4, pages 95:1–95:12, Juillet 2013. (Cité en page 27.)

[Lengyel 1998] Jed Lengyel. *The Convergence of Graphics and Vision*. Computer, vol. 31, no. 7, pages 46–53, Juillet 1998. (Cité en page 9.)

[Lessig 2010] Christian Lessig et Eugene Fiume. *On the Effective Dimension of Light Transport*. In Proceedings of the 21st Eurographics Conference on Rendering, EGSR'10, pages 1399–1403, Aire-la-Ville, Switzerland, Switzerland, 2010. Eurographics Association. (Cité en page 24.)

[Levin 2004] Anat Levin, Dani Lischinski et Yair Weiss. *Colorization Using Optimization*. ACM Trans. Graph., vol. 23, no. 3, pages 689–694, Août 2004. (Cité en page 56.)

[Levin 2006] Anat Levin, Dani Lischinski et Yair Weiss. *A Closed Form Solution to Natural Image Matting*. In Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1, CVPR '06, pages 61–68, Washington, DC, USA, 2006. IEEE Computer Society. (Cité en page 54.)

[Liu 2004] Yanxi Liu, Wen-Chieh Lin et James Hays. *Near-regular Texture Analysis and Manipulation*. In ACM SIGGRAPH 2004 Papers, SIGGRAPH '04, pages 368–376, New York, NY, USA, 2004. ACM. (Cité en page 54.)

[Lévy 2002] Bruno Lévy, Sylvain Petitjean, Nicolas Ray et Jérôme Maillot. *Least Squares Conformal Maps for Automatic Texture Atlas Generation*, 2002. (Cité en page 48.)

[Manay 2003] Siddharth Manay et Anthony Yezzi. *Anti-geometric diffusion for adaptive thresholding and fast segmentation.* IEEE Transactions on Image Processing, 2003. (Cité en page 52.)

[Marwah 2013] Kshitij Marwah, Gordon Wetzstein, Yosuke Bando et Ramesh Raskar. *Compressive Light Field Photography Using Overcomplete Dictionaries and Optimized Projections.* ACM Trans. Graph., vol. 32, no. 4, pages 46:1–46:12, Juillet 2013. (Cité en page 62.)

[Matusik 2003] Wojciech Matusik, Hanspeter Pfister, Matthew Brand et Leonard McMillan. *Efficient Isotropic BRDF Measurement.* In Proceedings of the 14th Eurographics Workshop on Rendering, EGRW '03, pages 241–247, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association. (Cité en pages 46 et 62.)

[Matusik 2009] Wojciech Matusik, Boris Ajdin, Jinwei Gu, Jason Lawrence, Hendrik P.A. Lensch, Fabio Pellacini et Szymon Rusinkiewicz. *Printing Spatially-Varying Reflectance.* ACM Trans. Graphics (Proc. SIGGRAPH Asia), vol. 28, no. 5, Décembre 2009. (Cité en page 7.)

[Mitra 2009] Niloy J. Mitra et Mark Pauly. *Shadow Art.* ACM Trans. Graph., vol. 28, no. 5, pages 156:1–156:7, Décembre 2009. (Cité en page 6.)

[Müller 2006] Pascal Müller, Peter Wonka, Simon Haegler, Andreas Ulmer et Luc Van Gool. *Procedural Modeling of Buildings.* In ACM SIGGRAPH 2006 Papers, SIGGRAPH '06, pages 614–623, New York, NY, USA, 2006. ACM. (Cité en page 6.)

[Munzner 2000] Tamara Macushla Munzner. *Interactive Visualization of Large Graphs and Networks.* PhD thesis, Stanford University, Stanford, CA, USA, 2000. AAI9995264. (Cité en page 5.)

[Needell 2010] Deanna Needell et Joel A. Tropp. *CoSaMP: Iterative Signal Recovery from Incomplete and Inaccurate Samples.* Commun. ACM, vol. 53, no. 12, pages 93–100, Décembre 2010. (Cité en page 62.)

[Neumann 1994] Laszlo Neumann, Martin Feda, Manfred Kopp et Werner Purgathofer. *A New Stochastic Radiosity Method for Highly Complex Scenes.* In Fifth Eurographics Workshop on Rendering, pages 195–206, Darmstadt, Germany, June 1994. (Cité en page 28.)

[Ngan 2005] Addy Ngan, Frédo Durand et Wojciech Matusik. *Experimental Analysis of BRDF Models.* In Proceedings of the Sixteenth Eurographics Conference

on Rendering Techniques, EGSR'05, pages 117–126, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association. (Cité en page 61.)

[Paillé 2012] Gilles-Philippe Paillé et Pierre Poulin. *SMI 2012: Full As-conformal-as-possible Discrete Volumetric Mapping.* Comput. Graph., vol. 36, no. 5, pages 427–433, Août 2012. (Cité en page 48.)

[Pajot 2011] Anthony Pajot, Loïc Barthe et Mathias Paulin. *Sample-Space Bright-Spot Removal Using Density Estimation (regular paper).* In Graphics Interface (GI), St John's, New Found Land (Canada), 25/05/2011-27/05/2011, pages 159–166, http://www.akpeters.com/, mai 2011. A K Peters. (Cité en page 27.)

[Peers 2009] Pieter Peers, Dhruv K. Mahajan, Bruce Lamond, Abhijeet Ghosh, Wojciech Matusik, Ravi Ramamoorthi et Paul Debevec. *Compressive Light Transport Sensing.* ACM Trans. Graph., vol. 28, no. 1, pages 3:1–3:18, Février 2009. (Cité en page 62.)

[Pohl 2013] Daniel Pohl, Gregory S. Johnson et Timo Bolkart. *Improved Pre-warping for Wide Angle, Head Mounted Displays.* In Proceedings of the 19th ACM Symposium on Virtual Reality Software and Technology, VRST '13, pages 259–262, New York, NY, USA, 2013. ACM. (Cité en page 8.)

[Prévost 2013] Romain Prévost, Emily Whiting, Sylvain Lefebvre et Olga Sorkine-Hornung. *Make It Stand: Balancing Shapes for 3D Fabrication.* ACM Transactions on Graphics (proceedings of ACM SIGGRAPH), vol. 32, no. 4, pages 81:1–81:10, 2013. (Cité en page 7.)

[Prusinkiewicz 1990] P. Prusinkiewicz et Aristid Lindenmayer. The algorithmic beauty of plants. Springer-Verlag New York, Inc., New York, NY, USA, 1990. (Cité en page 25.)

[Ren 2011] Peiran Ren, Jiaping Wang, John Snyder, Xin Tong et Baining Guo. *Pocket Reflectometry.* ACM Trans. Graph., vol. 30, no. 4, pages 45:1–45:10, Juillet 2011. (Cité en page 62.)

[Sanders 2010] Jason Sanders et Edward Kandrot. Cuda by example: An introduction to general-purpose gpu programming. Addison-Wesley Professional, 1st édition, 2010. (Cité en pages 41 et 44.)

[Sbert 1998] Mateu Sbert. *Random walk radiosity with infinite path length.* Computers & Graphics, vol. 22, no. 23, pages 161 – 166, 1998. (Cité en page 28.)

[Schwarzhaupt 2012] Jorge Schwarzhaupt, Henrik Wann Jensen et Wojciech Jaroz. *Practical Hessian-Based Error Control for Irradiance Caching.* ACM Transactions on Graphics, vol. 31, no. 6, November 2012. Article 193. (Cité en page 27.)

[Segovia 2006] B. Segovia, J. C. Iehl, R. Mitchaney et B. Peroche. *Bidirectional Instant Radiosity*. In Tomas Akenine-Moller et Wolfgang Heidrich, editeurs, Proceedings of Eurographics Symposium on Rendering (2006), pages 389–397, 2006. (Cité en page 27.)

[Seidel 1997] H.-P. Seidel, P. Slusallek et M. Stamminger. *Global Illumination Computations with Hierarchical Finite Element Methods*. In Proc. Mathematical Methods for Curves and Surfaces '97, 1997. Available from http://www9.informatik.uni-erlangen.de/eng/research/pub1997. (Cité en page 25.)

[Sen 2012] Pradeep Sen et Soheil Darabi. *On Filtering the Noise from the Random Parameters in Monte Carlo Rendering*. ACM Trans. Graph., vol. 31, no. 3, pages 18:1–18:15, Juin 2012. (Cité en page 27.)

[Shahrian 2013] Ehsan Shahrian et Deepu Rajan. *Using Texture to Complement Color in Image Matting*. Image Vision Comput., vol. 31, no. 9, pages 658–672, Septembre 2013. (Cité en page 54.)

[Sillion 1994] Francois Sillion et Claude Puech. Radiosity and global illumination. Morgan Kaufmann, San Francisco, CA, 1994. (Cité en page 24.)

[Silverman 1986] B.W. Silverman. Density Estimation for Statistics and Data Analysis. Chapman and Hall/CRC, 1 édition, Avril 1986. (Cité en pages 27 et 37.)

[Sloan 2002] Peter-Pike Sloan, Jan Kautz et John Snyder. *Precomputed Radiance Transfer for Real-time Rendering in Dynamic, Low-frequency Lighting Environments*. ACM Trans. Graph., vol. 21, no. 3, pages 527–536, Juillet 2002. (Cité en page 63.)

[Sloan 2008] Peter-pike Sloan. *Stupid Spherical Harmonics (SH) Tricks*, 2008. (Cité en page 50.)

[Smits 1994] Brian Smits, James Arvo et Donald Greenberg. *A Clustering Algorithm for Radiosity in Complex Environments*. In Computer Graphics Proceedings, Annual Conference Series, 1994 (ACM SIGGRAPH '94 Proceedings), pages 435–442, 1994. (Cité en pages 25 et 29.)

[StãNculescu 2013] Lucian StãNculescu, RaphaëLle Chaine, Marie-Paule Cani et Karan Singh. *SMI 2013: Sculpting Multi-dimensional Nested Structures*. Comput. Graph., vol. 37, no. 6, pages 753–763, Octobre 2013. (Cité en page 7.)

[Stam 1995] Jos Stam. *Multiple Scattering as a Diffusion Process*. In In Eurographics Rendering Workshop, pages 41–50, 1995. (Cité en page 38.)

[Thompson 2011] William Thompson, Roland Fleming, Sarah Creem-Regehr et Jeanine Kelly Stefanucci. Visual perception from a computer graphics perspective. A. K. Peters, Ltd., Natick, MA, USA, 1st édition, 2011. (Cité en page 9.)

[Veach 1997] Eric Veach et Leonidas J. Guibas. *Metropolis Light Transport.* In Computer Graphics (ACM SIGGRAPH '97 Proceedings), volume 31.3, pages 65–76, 1997. (Cité en pages 27, 58 et 63.)

[Vince 2010] John Vince. Introduction to the mathematics for computer graphics. Springer-Verlag, Berlin, Heidelberg, 3rd édition, 2010. (Cité en page 9.)

[Wang 2007a] Jue Wang, Maneesh Agrawala et Michael F. Cohen. *Soft Scissors: An Interactive Tool for Realtime High Quality Matting.* ACM Trans. Graph., vol. 26, no. 3, Juillet 2007. (Cité en page 54.)

[Wang 2007b] Kai Wang, Guillaume Lavoué, Florence Denis et Atilla Baskurt. *Three-dimensional Meshes Watermarking: Review and Attack-centric Investigation.* In Proceedings of the 9th International Conference on Information Hiding, IH'07, pages 50–64, Berlin, Heidelberg, 2007. Springer-Verlag. (Cité en page 7.)

[Wang 2009] Jiaping Wang, Peiran Ren, Minmin Gong, John Snyder et Baining Guo. *All-frequency Rendering of Dynamic, Spatially-varying Reflectance.* In ACM SIGGRAPH Asia 2009 Papers, SIGGRAPH Asia '09, pages 133:1–133:10, New York, NY, USA, 2009. ACM. (Cité en page 47.)

[Wei 2000] Li-Yi Wei et Marc Levoy. *Fast Texture Synthesis Using Tree-structured Vector Quantization.* In Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '00, pages 479–488, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co. (Cité en page 54.)

[Wei 2001] Li-Yi Wei et Marc Levoy. *Texture Synthesis over Arbitrary Manifold Surfaces.* In Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '01, pages 355–360, New York, NY, USA, 2001. ACM. (Cité en page 48.)

[Wilson 2004] Brett Wilson et Kwan-Liu Ma. *Rendering Complexity in Computer-generated Pen-and-ink Illustrations.* In Proceedings of the 3rd International Symposium on Non-photorealistic Animation and Rendering, NPAR '04, pages 129–137, New York, NY, USA, 2004. ACM. (Cité en page 6.)

[Wong 1998] Michael T. Wong, Douglas E. Zongker et David H. Salesin. *Computer-generated Floral Ornament.* In Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '98, pages 423–434, New York, NY, USA, 1998. ACM. (Cité en page 6.)

[Wright 2009] Stephen J. Wright, Robert D. Nowak et Mário A. T. Figueiredo. *Sparse reconstruction by separable approximation.* Trans. Sig. Proc., vol. 57, no. 7, pages 2479–2493, Juillet 2009. (Cité en page 62.)

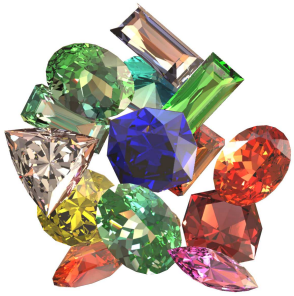
[Yoshida 2006] A. Yoshida, R. Mantiuk, K. Myszkowski et H.-P. Seidel. *Analysis of Reproducing Real-World Appearance on Displays of Varying Dynamic Range.* In Computer Graphics Forum, numéro 25 de CGF, pages 415¿–426, 2006. (Cité en page 6.)

[Zhou 2013] Qingnan Zhou, Julian Panetta et Denis Zorin. *Worst-case Structural Analysis.* ACM Trans. Graph., vol. 32, no. 4, pages 137:1–137:12, Juillet 2013. (Cité en page 7.)

Appendix B

Key publications

I'm including here a selection of four publications that are representative of my work. I mostly chose the ones I am the most proud of, in each of the sub-domains I referenced in this report (realtime rendering, geometry processing, image analysis, and simulation of radiative transfer) while trying to cover the full research period that is referred in this document.



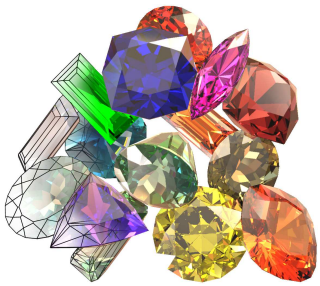
Graphics Gems Revisited

Fast and Physically-Based Rendering of Gemstones

Stephane Guy
PRIMA*

Cyril Soler
ARTIS *

GRAVIR/IMAG - INRIA



Abstract

We present an algorithm for rendering faceted colored gemstones in real time, using graphics hardware. Beyond the technical challenge of handling the complex behavior of light in such objects, a real time high quality rendering of gemstones has direct applications in the field of jewelry prototyping, which has now become a standard practice for replacing tedious (and less interactive) wax carving methods. Our solution is based on a number of controlled approximations of the physical phenomena involved when light enters a stone, which permit an implementation based on the most recent – yet commonly available – hardware features such as fragment programs, cube-mapping.

Keywords: Crystal optics, Hardware-based rendering, real time

1 Introduction

Gemstones are fascinating because they display many visually appealing phenomena thanks to their ability to alter light in a number of very specific ways. Examples include brilliance from internal reflections, fire due to dispersion of light, dichroism and doubling due to birefringence, color-shifting because of a camel-shaped absorbance spectrum, and darkening due to polarization. Unfortunately, the complexity of light interaction inside gemstones, due to their particular crystal structure, makes correct renderings of such objects very difficult to obtain.

Furthermore, our investigations among jewelry design packages show that computer aided prototyping has now become a standard [Doyle 2000]. Applications need a fast algorithm for rendering gemstones to allow the user to move and appreciate the variations in color and brightness over several views of the stone. The quality of the rendering is most important during the design of a piece hand-in-hand with a client who may want to see a high quality view of the expected result.

Up to now, the academic solutions which have been proposed for rendering gemstones have all been based on more or less complex ray tracing implementations, and therefore do not offer the possibility of real time display. To our knowledge, no commercial jewelry design software has such a capability. In the specifications of the two software packages *JewelSpace*™ (see `www.jewelspace.net`) and *JewelCAD*™ (see `www.jacadcam.com`), for instance, one learns that the first combines radiosity and ray tracing while the second uses OpenGL and ray tracing, which gives nice, but not instant, results.

We show in this paper that the convexity and polyhedral nature of faceted gemstones raises the possibility of an efficient hardware implementation, which we achieve by using the most recent hardware capabilities such as high level fragment programming [Mark et al. 2003], as well as more classical techniques such as cube mapping [Greene 1986]. It is indeed possible to rely on these tools for implementing pixel-based computation of Fresnel terms, directional light sampling and tone reproduction.

Our system introduces new possibilities of observing virtual gemstones: not only can the viewing conditions be changed in real time but also the physical properties (*e.g.,* color, refractive index) and even the geometry of the stone. Indeed, our management of rendering complexity offers the possibility to trade accuracy for speed, so as to maintain interactive rendering speed. Among applications, our algorithm could be a plugin for rendering faceted objects in a more complex scene as our results show. In terms of jewelry design, our contribution should be seen as a potential solution for real time examination of created objects with high rendering quality. For educational purposes it also permits to study interactively the visual properties of gemstone cuts in measured light conditions.

In section 3 we present a physical model for the behavior of light in colored gemstones. The importance of each phenomenon is then examined with a correct order of priority in terms of visual impact in Section 4, in order to derive a model suitable for hardware implementation. This implementation is presented in section 5. We finally give a number of results prior to concluding.

2 Related work

Yokoi *et al.* propose an algorithm [Yokoi et al. 1986] aimed particularly at the reproduction of asterism (*i.e.,* the effect seen on star sapphires) and chatoyancy effects. For this, a model for the dispersion of light rays into a distribution of microfacets is elaborated, which produces the expected result.

Yuan uses an adaptive ray tracing for adequately sampling the light spectrum for various wavelengths [Yuan et al. 1988], while keeping the overall complexity much lower than traditional dispersive ray tracing [Thomas 1986] and obtains fine images of diamonds. Dispersion is further investigated in [Sun et al. 2000b] and [Wilkie et al. 2000] for the rendering of glass-made objects.

Sun [Sun et al. 2000a] obtains images of colored diamonds using Fresnel reflection, volume absorption and light dispersion in a ray tracer. He also proposes a method to handle the non-linearity of absorption and the poor regularity of absorbance spectra using a composite spectral model.

These three papers constitute the few contributions in the field of computer graphics which aiming at rendering gemstones specifically. All these methods share a ray tracing basis with high computation time. None of them implements polarization which however incurs several visually important phenomena in gemstones.

Wolff proposes one of the first attempts to incorporate polarization effects in ray tracing [Wolff and Kurlander 1990], in the context

*Artis and PRIMA are teams within the GRAVIR/IMAG laboratory, a joint research unit of CNRS, INPG, INRIA, and UJF. {stephane.guy|cyril.soler}@imag.fr

of generalizing the Torrance-Sparrow reflectance model. For this, he uses the formalism of coherency matrices, introduced by Wolf in 1959 [Wolf 1959]. Another formalism (Stokes light vectors) was used by Wilkie in 2001 to get rid of complex numbers [Wilkie et al. 2001] and to incorporate polarization along with fluorescence effects in a ray tracer, with convincing results on semi-transparent objects. In 1994, Tannenbaum gave some details about the implementation of the birefringency phenomenon [Tannenbaum et al. 1994] for computing images of highly birefringent media such as calcite. As we explain below, birefringency is responsible for color variations in many stones.

3 Light propagation in gemstones

We start by presenting a physical model for light propagation in gemstones. This model will be used as a starting point for the approximations which lead to our simplified model (in Section 4), which is suitable for hardware implementation. It will also serve in a ray tracer (see Section 6) for validating these approximations.

Although the polarization state of light is, for most rendering applications, an expensive and unnecessary feature, it does play a critical role in the interaction of light with gemstones, and therefore must be incorporated into any rendering model that wants to reproduce these effects. Indeed, while the naked eye is not trained for detecting light polarization, the path of light through a faceted transparent object involves a number of selective changes in the polarization. This succession of changes is responsible for the darkening of some regions and, in the case of anisotropic crystal structures, a color change depending on the direction of propagation (Figure 3, *left* shows a combination of these two effects).

Let $\mathbf{E}$ be the electrical field of a monochromatic planar wave propagating along vector $\mathbf{s}$ at speed v and angular frequency ω. $\mathbf{E}$ is expressed in the plane orthogonal to $\mathbf{s}$ as space and time dependent 2D vector:

$$\mathbf{E}(\mathbf{r},t) = \begin{bmatrix} E^{\perp}\cos(\omega(t-\mathbf{r}\cdot\mathbf{s}/v)) \\ E^{\parallel}\cos(\omega(t-\mathbf{r}\cdot\mathbf{s}/v)+\delta) \end{bmatrix} = \mathcal{R}\begin{pmatrix} E^{\perp}e^{i(\omega(t-\mathbf{r}\cdot\mathbf{s}/v))} \\ E^{\parallel}e^{i(\omega(t-\mathbf{r}\cdot\mathbf{s}/v)+\delta)} \end{pmatrix} \quad (1)$$

$E^{\perp}$ and $E^{\parallel}$ are the amplitudes of each component and δ is the phase shift between the two components of $\mathbf{E}$. As shown, $\mathbf{E}$ may alternatively be represented as the real part of a complex-valued field. For rendering, we are interested in the intensity of $\mathbf{E}$, which is

$$I = {E^{\perp}}^2 + {E^{\parallel}}^2$$

3.1 Fresnel laws

Most crystal-structured materials are optically anisotropic (*e.g.,* tourmaline, sapphire, but not diamond). This comes from the geometric asymmetry of the atoms arranged in the structure, which favors different charge transfers – *i.e.,* different wavelength absorptions–, and different propagation speeds, depending on the direction of propagation. In the case of uniaxial crystals the medium is characterized by two indices of refraction n^o and n^e, and its optical axis $\mathbf{a}$. For a given wave propagation vector $\mathbf{s}$, we define the *crystal coordinate system* by its orthonormal basis X_a, Y_a, Z_a and the *principal plane* by the two vectors Y_a, Z_a:

$$\mathbf{X_a}(\mathbf{s}) = \frac{\mathbf{a}\times\mathbf{s}}{\|\mathbf{a}\times\mathbf{s}\|} \qquad \mathbf{Y_a}(\mathbf{s}) = \frac{\mathbf{s}\times(\mathbf{a}\times\mathbf{s})}{\|\mathbf{s}\times(\mathbf{a}\times\mathbf{s})\|} \qquad \mathbf{Z_a}(\mathbf{s}) = \mathbf{a} \quad (2)$$

Let ε be the dielectric tensor of the medium, defined by:

$$\varepsilon = {}^tO\begin{pmatrix} n^o & 0 & 0 \\ 0 & n^o & 0 \\ 0 & 0 & n^e \end{pmatrix} O \quad \text{with} \quad O = \begin{pmatrix} \mathbf{X_a}(\mathbf{s}) \\ \mathbf{Y_a}(\mathbf{s}) \\ \mathbf{Z_a}(\mathbf{s}) \end{pmatrix}$$

The dielectric displacement vector D of the wave is related to the electric field E by:

$$\mathbf{D} = \varepsilon\mathbf{E}$$

A light ray entering an anisotropic gemstone separates into two sub-rays linearly polarized along orthogonal directions (see Figure 1), called *ordinary ray*, or $o-$ray, and *extraordinary ray*, or $e-$ray [Born and Wolf 1999]. These rays belong to the only two categories of waves which are allowed to propagate in uniaxial crystals, and have their own characteristics:

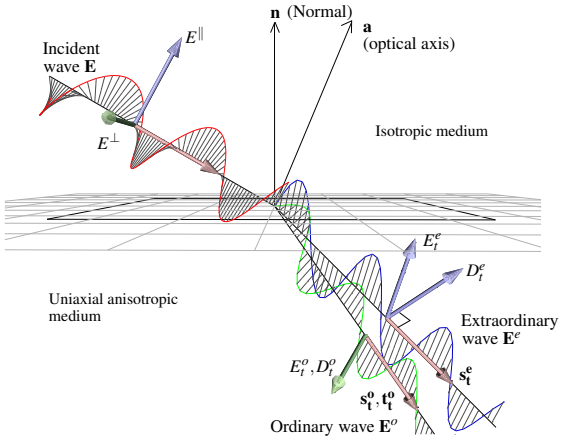


Figure 1: Formation of an ordinary and extraordinary ray when an incident light ray encounters an anisotropic medium.

The two waves have respective directions of propagation $\mathbf{s^o}$ and $\mathbf{s^e}$. The directions of polarization are such that for the $o-$ray, $\mathbf{D^o}$ vibrates perpendicular to the principal plane (*i.e.,*along $\mathbf{X_a}(\mathbf{s^o})$), and for the $e-$ray, $\mathbf{D^e}$ vibrates parallel to the principal plane (*i.e.,*along $\mathbf{Y_a}(\mathbf{s^e})$). The electric field of the $o-$ray is orthogonal to the direction of propagation $\mathbf{s^o}$ of its wave, whereas that of the $e-$ray is not, because $\varepsilon\mathbf{E^e}$ is not collinear to $\mathbf{D^e}$. The energy propagation vectors of the two waves (*i.e.,*the directions used for ray tracing) are thus $\mathbf{t^o} = \mathbf{s^o}$ and $\mathbf{t^e} \neq \mathbf{s^e}$.

The speed v of the two rays can be computed from the speed c of light in vacuum, using [Born and Wolf 1999]:

$$\begin{aligned} v^o &= \tfrac{c}{n^o} \\ v^e &= \left(\left(\tfrac{c}{n^o}\right)^2\cos^2\theta + \left(\tfrac{c}{n^e}\right)^2\sin^2\theta\right)^{\frac{1}{2}} \quad \text{with} \quad \cos\theta = \mathbf{a}\cdot\mathbf{s^e} \end{aligned}$$

While the $o-$ray lies in the plane of incidence and obeys Snell's law, the transmitted $e-$ray does not. Its ray direction $\mathbf{t_t^e}$ can be computed using the following formula [Beyerle and McDermid 1998] in which $\mathbf{n}$ is the interface normal and n_1 the refractive index of the incident medium:

$$\begin{aligned} &\mathbf{t_t^e} = \frac{\gamma^2(\mathbf{n'}^2\mathbf{s}+(R-\mathbf{n'}\cdot\mathbf{s'})\mathbf{n})}{\|\gamma^2(\mathbf{n'}^2\mathbf{s}+(R-\mathbf{n'}\cdot\mathbf{s'})\mathbf{n})\|} \quad \text{where} \quad \gamma = \tfrac{1}{n^o n^e}\varepsilon \quad \text{and} \\ &R = \left((\mathbf{n'}\cdot\mathbf{s'})^2 - \mathbf{n'}^2\mathbf{s'}^2 + \mathbf{n'}^2/n_1^2\right)^{\frac{1}{2}} \qquad \mathbf{n'} = \gamma\mathbf{n} \qquad \mathbf{s'} = \gamma\mathbf{s} \end{aligned} \quad (3)$$

Similar to refractions, internal reflections inside an anisotropic crystal splits light waves into ordinary and extraordinary rays. While the direction of the former obeys the classical rule of reflection, the direction of the $e-$ray obtained by internal reflection must be computed using [Beyerle and McDermid 1998]:

$$\mathbf{t_r^e} = \frac{\gamma^2(\mathbf{n'}^2\mathbf{s'} + 2(\mathbf{s'}\cdot\mathbf{n'})\mathbf{n'})}{\|\gamma^2(\mathbf{n'}^2\mathbf{s'} + 2(\mathbf{s'}\cdot\mathbf{n'})\mathbf{n'})\|} \quad (4)$$

Finally, for a given wave refracting and reflecting at the interface between the air and an anisotropic medium, we need to express the Fresnel coefficients which permit us to compute the refracted and reflected fields. This happens when light enters or exits a gemstone, and depending on the case, reflected or refracted waves may either be the sum of an ordinary and extraordinary rays, or a unpolarized

wave. We treat both cases at once by considering the interface between two anisotropic media. If one medium is isotropic, the waves in this medium are still the sum of two orthogonally polarized wave components and can thus be represented as a '$o-$ray' and a '$e-$ray' with the same direction of propagation with arbitrary choice of the 'optical axis' of the medium. We take the convention that subscripts **t** and **r** stand for *transmitted* and *reflected* fields, while **o** and **e** stand for *ordinary* and *extraordinary*.

Let $\mathbf{E_i}$ be an incident linearly polarized wave of amplitude E_i, vibrating direction $\mathbf{e_i}$, and speed $v_i = \frac{c}{n_i}$. We want to find the coefficients $\alpha_r^o, \alpha_r^e, \alpha_t^o, \alpha_t^e$ by which to multiply E_i to obtain the amplitudes of the four reflected and refracted fields.

Maxwell theory requires that the tangential components of the total electric field and the magnetic vector $\mathbf{H} = \frac{c}{\mu v}\mathbf{s} \times \mathbf{E}$ be continuous across the surface (where μ is the magnetic permissivity, supposed identical in both media):

$$\begin{aligned}\mathbf{n} \times (\mathbf{E_i} + \mathbf{E_r^o} + \mathbf{E_r^e}) &= \mathbf{n} \times (\mathbf{E_t^o} + \mathbf{E_t^e}) \\ \mathbf{n} \times (\mathbf{H_i} + \mathbf{H_r^o} + \mathbf{H_r^e}) &= \mathbf{n} \times (\mathbf{H_t^o} + \mathbf{H_t^e})\end{aligned}$$

Let $\mathbf{e_r^o}, \mathbf{e_r^e}, \mathbf{e_t^o}, \mathbf{e_t^e}$ and $\mathbf{h_r^o}, \mathbf{h_r^e}, \mathbf{h_t^o}, \mathbf{h_t^e}$ be the respective vibrating directions of the electrical and magnetic fields. By expressing each component $\mathbf{E_k^l}$ (resp. $\mathbf{H_k^l}$) as $\alpha_k^l E_i \mathbf{e_k^l}$ (resp. $\alpha_k^l E_i n_k^l \mathbf{h_k^l}$, with $\mathbf{h_k^l} = \mathbf{s_k} \times \mathbf{e_k^l} / \|\mathbf{s_k} \times \mathbf{e_k^l}\|$), and computing the dot product of these equations with two independent vectors $\mathbf{v_1}$ and $\mathbf{v_2}$ in the interface plane, one obtains [C.McClain et al. 1993]:

$$\begin{bmatrix} -\mathbf{v_1}\cdot\mathbf{e_r^o} & -\mathbf{v_1}\cdot\mathbf{e_r^e} & \mathbf{v_1}\cdot\mathbf{e_t^o} & \mathbf{v_1}\cdot\mathbf{e_t^e} \\ -\mathbf{v_2}\cdot\mathbf{e_r^o} & -\mathbf{v_2}\cdot\mathbf{e_r^e} & \mathbf{v_1}\cdot\mathbf{e_t^o} & \mathbf{v_1}\cdot\mathbf{e_t^e} \\ -n_r^o\mathbf{v_1}\cdot\mathbf{h_r^o} & -n_r^e\mathbf{v_1}\cdot\mathbf{h_r^e} & n_t^o\mathbf{v_1}\cdot\mathbf{h_t^o} & n_t^e\mathbf{v_1}\cdot\mathbf{h_t^e} \\ -n_r^o\mathbf{v_2}\cdot\mathbf{h_r^o} & -n_r^e\mathbf{v_2}\cdot\mathbf{h_r^e} & n_t^o\mathbf{v_2}\cdot\mathbf{h_t^o} & n_t^e\mathbf{v_2}\cdot\mathbf{h_t^e} \end{bmatrix} \begin{bmatrix} \alpha_r^o \\ \alpha_r^e \\ \alpha_t^o \\ \alpha_t^e \end{bmatrix} = \begin{bmatrix} \mathbf{v_1}\cdot\mathbf{e_i} \\ \mathbf{v_2}\cdot\mathbf{e_i} \\ n_i\mathbf{v_1}\cdot\mathbf{h_i} \\ n_i\mathbf{v_2}\cdot\mathbf{h_i} \end{bmatrix} \quad (5)$$

This linear system can be solved numerically. In the case of total reflection it should first be simplified into a $2x2$ system by suppressing the α_t^o and α_t^e unknowns. Up to now $\mathbf{E_i}$ has been supposed linearly polarized, which applies for $e-$rays and $o-$rays inside an anisotropic medium. If the incident medium is isotropic, the two solutions corresponding to setting $\mathbf{e_i}$ orthogonal and then parallel to the incident plane gives the 8 needed coefficients. If both media are chosen isotropic, and if the 'optical axis' is set to be the normal of the interface, we have checked that the obtained solution corresponds to the well known Fresnel formulas given in Figure 2.

$$\begin{aligned} F_r^\perp &= \alpha_r^{oo} = -\sin(\theta_i - \theta_t)\sin(\theta_i + \theta_t)^{-1} \\ F_r^\parallel &= \alpha_r^{ee} = \tan(\theta_i - \theta_t)\tan(\theta_i + \theta_t)^{-1} \\ F_t^\perp &= \alpha_t^{oo} = 1 + F_\perp^r \\ F_t^\parallel &= \alpha_t^{ee} = \left(1 - F_\parallel^r\right)\cos\theta_i \cos\theta_t^{-1} \end{aligned}$$

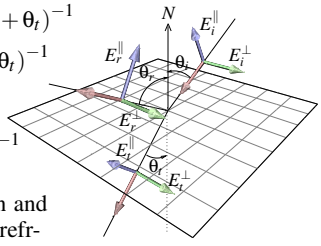


Figure 2: Geometric configuration and Fresnel coefficients for a light ray refracting and reflecting at the interface between two isotropic media. The crossed coefficients $\alpha_r^{oe}, \alpha_r^{eo}, \alpha_t^{oe}$ and α_t^{eo} are null in this particular case.

3.2 Absorption

As a general rule, a fraction of the light traveling inside a transparent material is absorbed. The absorption along a path (x_0, x) is ruled by the *Bouguer-Lambertian* law [Wyszecki and Stiles 1982], giving the resulting intensity after a distance d:

$$I_\lambda(x) = I_\lambda(x_0)e^{-\kappa(\lambda)d} \quad (6)$$

κ is called *the absorbance* of the medium and depends on the wavelength. Absorption is responsible for the color of gemstones, and the absorbance spectrum acts as a tracer for the chemical nature, geographic source and applied treatments of a stone.

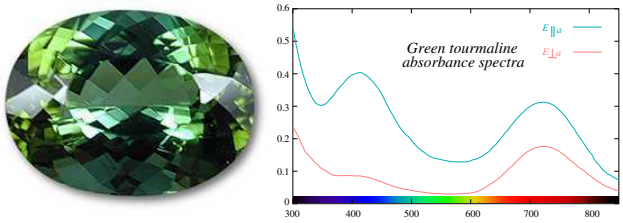


Figure 3: *left:* Photo of pleochroism in a green tourmaline. The optical axis is left-to-right oriented. Rays entering on the left side perpendicular to the page have most of their path parallel to the optical axis before they come out to the eye from the right side. They are thus mainly attenuated by the $E_\parallel$ (mainly green) spectrum. *Photo: Wimon Manorotkul/Pala International (www.palagems.com); used with permission. Right:* absorbance spectra of tourmaline for waves components polarized along/perpendicular to the optical axis.

Here again the optical anisotropy of gemstones plays an important role: because the $o-$rays and $e-$rays have different polarization directions, they are absorbed differently by the stone. The following formula [Born and Wolf 1999] gives the absorbance for the two rays:

$$\begin{aligned} \kappa^o &= K^o \\ \kappa^e &= K^o\cos^2\theta + K^e\left(\tfrac{n^o}{n^e}\right)^2\sin^2\theta \end{aligned} \quad (7)$$

In this formula, K^o and K^e are characteristic constants of the medium, and $\cos\theta = \mathbf{s^e} \cdot \mathbf{a}$ is the cosine of the angle between the extraordinary wave propagation direction and the optical axis of the stone.

Depending on the angle with which a ray enters a birefringent gemstone, the refracted $e-$ray will have variable direction; its absorption and color contribution to the outgoing light will consequently vary. This phenomena, called *pleochroism*, is illustrated on Figure 3 for a tourmaline gemstone.

Many gemstones display this behavior (*e.g.,*sapphires, rubies, tourmalines [Hughes 1997; Nassau 2001]). Other gemstones (like andalusite) display three distinct colors. This comes from their crystal structure, which has two optical axes instead of one. The computation of absorption in this context is beyond the scope of this paper. Some of these stones, however, behave visually as if they were uniaxial (*e.g.,*peridot and andalusite) because two of the three absorbance spectra are nearly identical.

The model presented above represents the state of the art in the understanding of light transport in gemstones, except for two deliberate approximations: First, the model treats wavelengths independently, and therefore cannot represent fluorescence in gemstones. Although there is fluorescence in some gems (e.g., some sapphires, synthetic diamonds), it is weak and mainly affects UV light outside the visible spectrum, so we ignore it. Second, in absorption, the polarizations of the *e*-ray and *o*-rays are in fact very slightly elliptical rather than strictly linear [Born and Wolf 1999]; we nonetheless treat them as linear.

4 Adopted model

At this point of the paper, we discuss the importance and priorities that should be given to the phenomena previously described, keeping in mind a possible hardware implementation. In section 4.1 we justify our choices for representing color as three separate wavelengths, and examine in section 4.2 how light will be represented for each wavelength.

4.1 Representation of color

Choosing a good representation of color is usually a compromise between visual accuracy of the result and a bulky set of coefficients.

Because the constraints of the hardware, we have chosen to work with 3 color components. Richer spectral rendering [Peercy 1993] could still be achieved by adding more passes and a final reconstruction, at the expense of rendering time [Sun et al. 1999].

We thus limit ourselves to spectral sampling, although the non linear relationship between absorbance and transmittance tends to saturate colors when darkening them [Sun et al. 1999]. Such an approximation on the facility to approach the attenuation by its linear counterpart is justified below.

One other important aspect of spectral sampling is to correctly choose the absorbance coefficients κ_r, κ_g and κ_b for the R, G and B channels. Directly reading spectral absorption coefficients from the spectral absorbance curve at the exact wavelengths of red ($700.0nm$), green ($546.1nm$) and blue ($435.8nm$) introduces significant errors due to peaks in the absorbance curve [Sun et al. 1999]. We thus need a way to extract significant enough absorbance values from the stone's spectral absorbance curves. Inspired by existing approximations for reflectance [Borges 1991], we propose the following:

The R, G, B color components perceived by human eye for a given spectrum S are computed using the color matching functions $\overline{r}$, $\overline{g}$ and $\overline{b}$ of λ [Wyszecki and Stiles 1982] by:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \int_\lambda \begin{bmatrix} \overline{r}(\lambda) \\ \overline{g}(\lambda) \\ \overline{b}(\lambda) \end{bmatrix} S(\lambda) d\lambda$$

Following Equation 6, a spectrum S_0 corresponding to white light will transform, after a path of length x, into

$$S(x,\lambda) = S_0(\lambda) e^{-\kappa(\lambda)x} \quad e.g., \quad R(x) = \int_\lambda \overline{r}(\lambda) S_0(\lambda) e^{-\kappa(\lambda)x} \quad (8)$$

We are looking for an absorbance value κ_r such that $R(x) = R(0)e^{-\kappa_r x}$ approximates equation 8 for small values of x. We thus take:

$$\kappa_r = -\frac{1}{x} \ln \frac{R(x)}{R(0)} = -\frac{1}{x} \ln \left[\frac{1}{R(0)} \int_\lambda S_0(\lambda) \overline{r}(\lambda) e^{-\kappa(\lambda)x} d\lambda \right]$$

which, for small distance values x, is approximated by:

$$\kappa_r = \frac{1}{R_0} \int_\lambda \kappa(\lambda) \overline{r}(\lambda) S_0(\lambda) d\lambda \quad \text{using} \quad R_0 = \int_\lambda \overline{r}(\lambda) S_0(\lambda) d\lambda \quad (9)$$

Proceeding identically for the green and blue components we obtain suitable absorbance coefficients from the absorbance spectra, while avoiding artifacts of direct sampling of absorbance functions with peaks.

The error of the above approximation for $\kappa_r, \kappa_g, \kappa_b$ depends on the extent to which absorption differs from its linear approximation in Equation 9. The error is thus small because the diameter L of the stone, times the absorbance κ is small (the absorbance is computed piecewise between successive internal reflections of light). For typical values of $L = 1$cm and $\kappa = 0.4$ for instance, we get an error of $\frac{e^{-\kappa L} - (1-\kappa_L)}{e^{-\kappa L}} = 0.00125\%$

4.2 Representation of monochromatic light

We adopt the formalism of *coherency matrices* [Glassner 1995; Wolf 1959] for representing the intensity and the polarization state of the electric field along a ray of light. The coherency matrix of a field $\mathbf{E} = \left(e^\perp(t), e^\parallel(t)\right)$ is defined as

$$J = \begin{bmatrix} < e^\perp e^{\perp *} > & < e^\perp e^{\parallel *} > \\ < e^\parallel e^{\perp *} > & < e^\parallel e^{\parallel *} > \end{bmatrix} = < E\, E^* > = \begin{bmatrix} J_{xx} & J_{xy} \\ J_{yx} & J_{yy} \end{bmatrix}$$

where E is the complex representation of $\mathbf{E}$, E^* is the conjugate transpose of E, and $< u >$ denotes the mean value of u over time. The intensity of the field is given by $I = J_{xx} + J_{yy}$. From this, the coherency matrix of an incoherent (non polarized) light ray of intensity I_0 is [Wolf 1959]:

$$J_{\text{incoherent}} = \frac{1}{2} I_0 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

For any linear transformation M (also called a *modifier matrix*) applied to $\mathbf{E}$, the corresponding coherency matrix becomes, from the definition of J, $J' = MJM^*$. It is thus possible to compute the matrix J along a ray by applying successive modifier matrices corresponding to refraction and internal reflections on the faces of the gemstone, and rotations to account for the change in coordinate systems between two successive media interfaces. The matrices involved for refraction, reflection and rotation with an angle of θ, are respectively:

$$M^t = \begin{bmatrix} \alpha_t^{oo} & \alpha_t^{oe} \\ \alpha_t^{eo} & \alpha_t^{ee} \end{bmatrix} \quad M^r = \begin{bmatrix} \alpha_r^{oo} & \alpha_r^{oe} \\ \alpha_r^{eo} & \alpha_r^{ee} \end{bmatrix} \quad R_\theta = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

In birefringent gemstones, the direction of the ordinary ray is easily computed using Snell's law, but the intervention of a different direction for the extraordinary refracted and reflected rays make the computation cost grow exponentially with the depth of paths inside the stone. We computed the maximum angle between the $o-$ray and $e-$ray for all incidence angles in $[0, \pi/2]$ and any orientation of the optical axis, for common gemstones (zircon being an extreme case):

Material	Zircon	Tourmaline	Sapphire	Emerald
Angle (deg.)	2.328	0.767	0.45	0.324
Error	4.06 %	1.34 %	0.785 %	0.565 %

The "error" field gives the distance between the two images of an edge on the far side of the stone, as a fraction of the total size of the stone. For a 1cm Zircon, for instance, the two images would be 0.5mm apart; on a 1000-pixel image, this would 50 pixels, but this is the extreme case (by far). The validity of this approximation is confirmed by the photos of the tourmaline gemstones on Figures 3 and 11: no doubling of the edges seen by refraction is perceptible whereas in the orientation of the optical axis corresponds to a case of maximum deviation in Figure 3.

Computing Fresnel coefficients using Equation 5 can not conveniently be implemented in graphics hardware. Contrarywise, Fresnel coefficients at the interface of two isotropic media (given in Figure 2) can be tabulated very efficiently. Because $o-$rays and $e-$rays propagate along close directions and have orthogonal polarization, the generalized Fresnel coefficients are very close to the isotropic ones when the difference $n_e - n_o$ is small, if expressed into the same coordinate systems. Let $R_{\theta_i}, R_{\theta_r}, R_{\theta_t}$ be the rotation matrices which align the coordinate systems for the isotropic Fresnel coefficients of Figure 2 to the corresponding implicit coordinate systems of the general Fresnel coefficients of equation 5 in the coordinate system on the principal planes (Equation 2). We have:

$$M^t \approx R_{\theta_i} \begin{bmatrix} F_\parallel^t & 0 \\ 0 & F_\perp^t \end{bmatrix} R_{\theta_t} \quad \text{and} \quad M^r \approx R_{\theta_i} \begin{bmatrix} F_\parallel^r & 0 \\ 0 & F_\perp^r \end{bmatrix} R_{\theta_r} \quad (10)$$

Equality holds when $n^o = n^e$. For instance let's consider the case of a ray entering an anisotropic medium of optical axe $\mathbf{a}$ from an isotropic medium. $\mathbf{n}$ is the interface normal and $\mathbf{s_t}$ the propagation vector of the transmitted wave. Generalized Fresnel coefficients already relate the reflected field to the incident field in the same coordinate system than in the isotropic case, but the transmitted field is expressed in the coordinate system with the orthogonal vibrating direction given by $\mathbf{a} \times \mathbf{s}$, so one should take:

$$\theta_i = 0, \quad \theta_r = 0, \quad \cos\theta_t = \mathbf{n} \times \mathbf{s_t} \cdot \mathbf{a} \times \mathbf{s_t}, \quad \sin\theta_t = \mathbf{n} \times \mathbf{s_t} \cdot \mathbf{s_t} \times (\mathbf{a} \times \mathbf{s_t})$$

The proposed approximation works very well, even on zircon, as illustrated on figure 4.

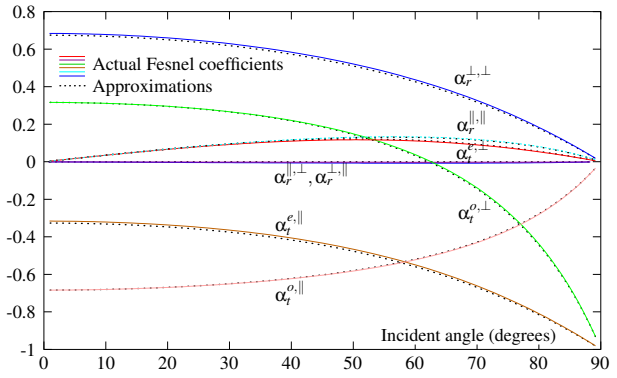


Figure 4: The colored curves present the 8 Fresnel coefficients $\alpha_{t|r}^{[o|e][o|e]}$ at the interface between the air and a birefringent medium (zircon). The black-dotted curves represent their approximations obtained through equations 10. The interface is the plane $z=0$, the optical axis is $\mathbf{a}=(\cos(\pi/6),0,\sin(\pi/6))$ and the incidence plane is chosen so as to correspond to the case of highest deviation between the $o-$rays and the $e-$rays.

Contrarywise, the change in absorption due to the polarization of rays with respect to the optical axis of the stone dramatically affects the rendering color and should not be neglected (see Figure 3).

Because the waves along the $o-$ray and the $e-$ray are polarized orthogonally to each other, we can represent them using a single coherency matrix in the coordinate system based on the ordinary and extraordinary vibration directions and their – supposed shared – direction of propagation $\mathbf{s}$. In this coordinate system, the matrix of a wave propagating inside the crystal is therefore diagonal, and is attenuated by a diagonal modifier matrix depending on the direction of propagation, using the attenuation coefficients of Equation 7:

$$A(s) = \begin{bmatrix} e^{-\kappa^o(s)l} & 0 \\ 0 & e^{-\kappa^e(s)l} \end{bmatrix} \tag{11}$$

In the case of isotropic crystals (*e.g.,* diamonds, garnets) the matrix $A(s)$ becomes identity times the attenuation given by Equation 6.

5 Rendering algorithm

Figure 5 shows the path of the light obtained by tracing a ray from the eye to a point on the stone. To compute the resulting intensity J_0 along such a path, we need to add the contributions of light at each interface P_k between the stone and the air, and accounting for attenuation $A_{k\to k+1}$ along segments $[P_k,P_{k+1}]$ inside the stone. Denoting by J_k the coherency matrix of the light from point P_k in the path, we have:

$$\begin{array}{rcl} J_0 & = & M_0^r I_0^i M_0^{r*} + M_0^t A_{0\to 1} \quad J_1 \quad A_{0\to 1}^* M_0^{t*} \\ J_k & = & M_k^t J_k^i M_k^{t*} + M_k^r A_{k\to k+1} \, J_{k+1} \, A_{k\to k+1}^* M_k^{r*} \end{array} \tag{12}$$

For rendering a gemstone using classical ray tracing, one would collect these contributions from back to front (*i.e.,* at P_n,P_{n-1} and finally P_0), transforming rays at each successive interface to account for refraction, external reflection, or internal reflections encountered along the path.

Our hardware-based algorithm relies on the fact that, for a given depth k, the set of points P_k that contribute to the image through the same succession of transformations, can be rendered at the same time using a fragment program. Such a set is called a *facet*. We regroup facets in a tree structure called the *facet tree*. Each node of the facet tree at depth k contains a product of k transformations and a region of points included in one face of the gemstone.

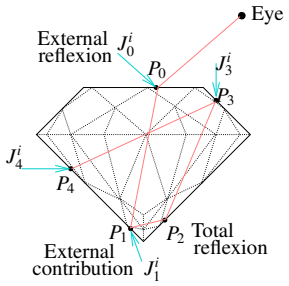


Figure 5: Different light intensities $I_{0,1,2,3,4}^i$ contribute to the final intensity at point P_0 along incident directions associated to external reflections (at P_0) and internal refraction (at P_1,P_3,P_4). The ray is followed until the attenuation lowers significantly the chances of missing a hotspot in the incoming light.

Rendering a gemstone from a given viewpoint thus requires for each frame to (1) build the facet tree corresponding to current viewpoint and (2) accumulate contributions of all facets of the facet tree, from back to front, using a fragment program. The lighting environment used for rendering gemstones is stored in a cubemap [NVIDIA Corporation 2000]. As illustrated in the result section, rendering gemstones needs to deal with high dynamic range lighting and rendering. All computations are thus performed with floating point values and a final tone reproduction pass (3) turns the result of pass (2) it into a $8-$bits rgb image. Pass $(1),(2)$ and (3) are detailed in Sections 5.1,5.2 and 5.3.

When changing the viewpoint the facet tree changes and must therefore be updated. This means that not only the viewpoint, but also the geometry of the gemstone as well as its physical parameters (refractive indices, orientation of optical axis, attenuation) can arbitrarily be changed during rendering, at the same cost.

5.1 Pass 1: construction of the facet tree

Because refraction is not in general a linear transform, the facets as defined above are not polygons. However the nonlinearity does not noticeably affect the refracted images of short segments such as the edges of a gemstone, as shown on figure 6, and we adopt, for representing refraction through a front face of the gemstone, the linear approximation proposed by Heckbert for beam tracing [Heckbert and Hanrahan 1984]. To each facet is thus associated a fictive viewpoint. Note however, that when we render the facet tree, the refraction direction of incoming light I_k at point $P_{k\neq 0}$ will be computed exactly by the fragment shader. Approximations of refraction only affect the point P_0.

Figure 6: *Left*: image computed using exact refraction with our ray tracer. *Center*: image computed with graphics hardware algorithm and linearized refraction. *Right*: difference image.

Thanks to this approximation and to the linearity of internal reflections on the gemstone faces, each facet is an actual polygon and a subset of a single face of the gemstone. We compute facets with the OpenGL feedback buffer using the following algorithm:

At level 0 of the tree the facets are the faces of the gemstone polyhedron directly seen from the viewpoint. At level 1 the child facets of level 0 facet f_i^0 correspond to the intersection of a beam traced from the viewpoint through f_i^o with the gemstone transformed by refraction. At subsequent levels the gemstone is further transformed

by reflection through the support face of the parent facet and clipped with this facet.

The window coordinates of each facet are computed by rendering the initial geometry of the gemstone (stored in a *display list*) into the *OpenGL* feedback buffer [Neider et al. 1993], using the appropriate viewpoint and transformation, while inserting clipping planes corresponding to the edges of the parent facet as shown on Figure 7.

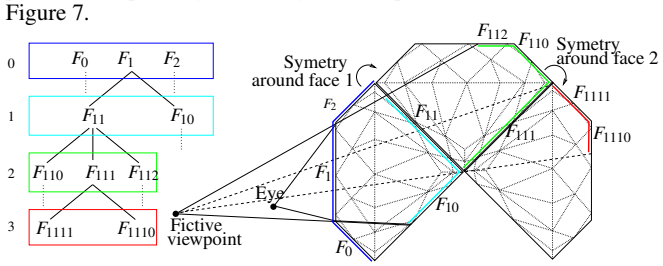
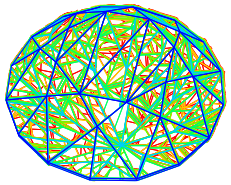


Figure 7: Construction of the *facet tree*, using a fictive viewpoint for linearized refraction and successive symmetries for internal reflections. For level k the gemstone model is transformed through reflection/refraction around the support plane of the parent facet, then drawn in the OpenGL feedback buffer, and clipped by the edges of the parent facet. The resulting mesh is displayed on the right, using the same color for facets of the same level.



To minimize costly access to the feedback buffer, the facet tree is constructed breadth-first. The depth of the tree is limited by three conditions: (1) the total path length to a facet attenuates light strongly enough to reduce the resulting intensity under a fixed value; (2) the area of the facet is smaller than a user defined threshold; (3) the computation time down to the current level exceeds the frame-rate limit.

Because refraction indices depend on wavelengths, the facet tree should not be the same for the red, green and blue components. Depending on the intensity of dispersion, we compute either one single facet tree corresponding to an average refractive index, or three distinct facet trees, at the expense of frame rate. However, the fragment shader which computes the refraction of light at points $P_{1,\ldots,n}$ still uses the correct indices of refraction. We thus still achieve in the worst case an approximation of dispersion effects, as shown on Figure 8.

Figure 8: Image of a highly dispersive stone, computed using exact refraction with our ray tracer(*top left*) and with our hardware algorithm with one facet tree (*top right*). *Left*: a difference image shows that some –but not all, as explained in the text– rainbow effects are missed by our algorithm in this case.

For a given maximum depth, fully building the facet tree is not in fact necessary: during this operation, we estimate the cumulative effect of attenuation and Fresnel coefficients for a single point on each facet down to the next level. We locally prune the construction if the contribution of next facet will be less than a threshold in percentage to the accumulated energy before the current facet. Section 6.2 shows a practical example of this.

5.2 Pass 2: rendering the facet tree

We implemented and tested the rendering algorithm of the facet tree on a *NVidia GeForceFX* 5900 using *Cg* programming [Lindholm et al. 2001].

Figure 9 summarizes our implementation: the facet tree is traversed breadth-first, by decreasing order of level. At a given level $k > 0$, a $P-$buffer is filled with the result of the *internal reflection* fragment program, which computes the exact refraction of light entering the gemstone from the cube-map, as well as the internal reflection of the contribution of level $k+1$ stored in the accumulation buffer. The combined result is copied to the accumulation buffer for level $k-1$. At level 0 the *external reflection* fragment program is used instead, for adding the result of the previous calls seen by refraction, with the light reflecting on the faces directly visible. Both fragment programs compute the attenuation using Equations 7 and 11. The path length is obtained by subtracting the distance from the fictive viewpoint to the current pixel (*c.f.* Figure 7) to that of the pixels in the previous result. The Fresnel coefficients are computed using equation 10, from the formulas of Figure 2, tabulated in 1*D* float textures.

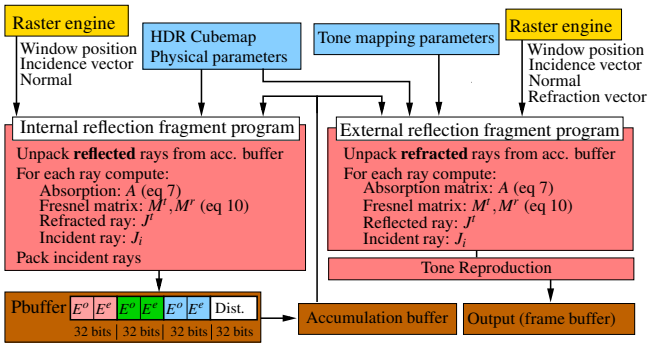


Figure 9: Implementation using *Cg* programming. See text.

The incoming light is represented using a HDR cube-map composed of three traditional $4 \times 8-$bits cube-maps, one per color channel, each representing $32-$bits fixed point values. The computation by the fragment programs is performed using floating point numbers. We therefore can not simply blend the contributions of facets into the $(8-$bits RGBA) frame buffer. This justifies the use of a $32-$bits floating point RGBA $P-$buffer. Fresnel relationships being always expressed in coordinate systems were the matrices are diagonal, we only need to store two components per channel (E^o, E^e for anisotropic media, resp. $E^{\perp}$, $E^{\parallel}$ for isotropic ones). Each component being represented as a $16-$bits *half float* [Bogart et al. 2003], one can fit in the *RGB* channels of the $P-$buffer the electric fields for red,green and blue. This encoding requires to *un-pack* the values before using them. The A channel contains the distance of the pixel to the fictive viewpoint at the last treated level in the facet tree.

5.3 Pass 3: tone reproduction

Tone reproduction is achieved entirely in hardware using a publicly available shader of Industrial Light+Magic [Bogart et al. 2003], to display floating point images stored in the *OpenEXR* format. Such an approach makes comparisons between ray-traced images, pictures of gemstones, and images rendered using the hardware method very convenient, since the first two are produced in OpenEXR format and displayed using the same shader as the one used in the hardware implementation. We also tested hardware based glare effect [Spencer et al. 1995; Debevec 1998] on our HDR images with success, as in the teaser on page 1, and the diamond on Figure 11. Glare effect was not used anywhere else in the paper, to prevent masking important artifacts and details of the method.

6 Results

6.1 *Modus operandi*

The work presented may seem hard to reproduce without a clear road map, which we give here.

Using the reference ray tracing algorithm requires working with coherency matrices. Directions of the ordinary reflected and transmitted $o-$rays can be computed using Snell's law, while those of the $e-$rays are given by Equations 3 and 4. Absorption is computed by Equations 7 and 11. General Fresnel coefficients at each interface are obtained by solving the $4x4$ linear system in Equation 5.

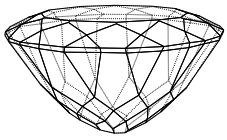
Our hardware based algorithm differs from a standard ray tracer in that ray directions always follow Snell's law; Fresnel coefficients are obtained using Equation 10, and refraction is linearized as explained in Section 5.1.

Both methods use equation 12 for composing coherency matrices at media interfaces, and use the same set of parameters, as described in the following three paragraphs:

System parameters We detail in the table below the parameters we used for our experiments. One needs the nature of the medium ('I'=isotropic, 'U'=birefringent uniaxial, 'B'=biaxial) and the refractive indices n^o and n^e when applicable. Values are indicated for the green channel whereas red and blue are obtained after adding or subtracting the value after '$\pm$' (half the *dispersion* value found in usual tables). Tricolored absorbance values are computed using Equation 9 from the absorbance spectra of each stone[1].

Gemstone	Type	$(K_r, K_g, K_b)_{o\|e}$	Colors	n_o,n_e
Garnet	I	$(0.136, 0.153, 0.175)$	orange-red	$1.730 \pm .014$
Tourmaline	U	$(0.033, 0.034, 0.082)_o$	yellow green	$1.642 \pm .011$
(Dravite)		$(0.010, 0.076, 0.015)_e$	blue green	$1.619 \pm .011$
Peridot	B	$(0.023, 0.015, 0.051)_o$	green	$1.680 \pm .010$
		$(0.011, 0.003, 0.028)_e$	yellow green	$1.640 \pm .010$
Diamond	I	$(0.001, 0.001, 0.001)$	white	$2.410 \pm .022$
Sapphire	U	$(0.332, 0.270, 0.156)_e$	light blue	$1.768 \pm .009$
		$(0.165, 0.147, 0.185)_o$	violetish blue	$1.760 \pm .009$
Andalusite	B	$(0.0056, .006, .0183)_o$	greenish red	$1.635 \pm .005$
		$(0.170, 0.175, 0.257)_e$	yellowish green	$1.644 \pm .005$

Geometric models Models of standard gemstone cuts are readily available on the internet[2]. However, problems arise when a particular gemstone needs to be simulated, as in our comparisons with photographs. While expensive gemstones (*e.g.,* very clear and fine quality stones) tend to be cut using very precise standards, more affordable pieces often display an *ad-hoc* cut so as to respect constraints such as avoiding inclusions. Laser-scanning gemstones is not applicable due to both their size and specularity. We also tried $X-$ray tomography with very poor results.



The solution we used is based on manually designing the mesh and applying an iterative relaxation algorithm on vertex positions so as to fit symmetries, distance and planarity constraints measured on the stones. This works well provided that the hand-designed mesh is not too far from the real object. The model *at right* was designed that way and corresponds to the tourmaline of Figure 11.

Acquisition of light We used traditional methods for acquiring cube-maps. Photographs of a mirrored ball were taken along two orthogonal directions and using a set of 10 different exposures. Each set is first converted into a HDR image in Radiance format. The two HDR images are combined using HDRShop™ (`www.debevec.com/HDRShop`), and the result saved as a cube-map. We proceed identically to obtain HDR images of sample gemstones in the same lighting environment.

1 - *e.g.,* at `http://minerals.gps.caltech.edu/FILES/Visible/`
2 - *e.g.,* at `http://www.3dlapidary.com/`

6.2 Additional validation tests

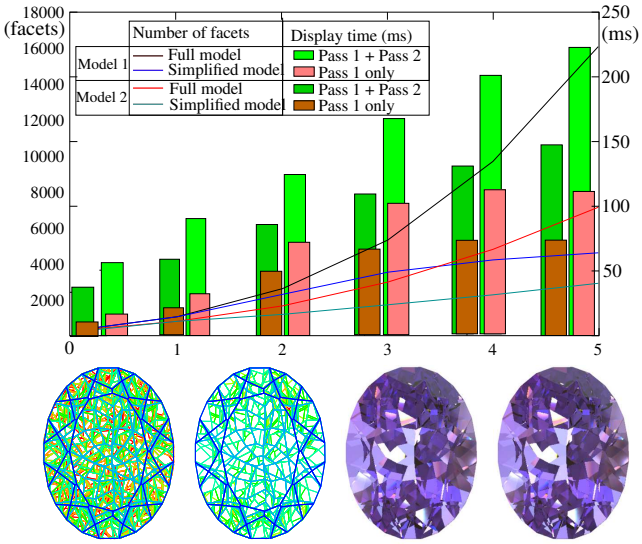


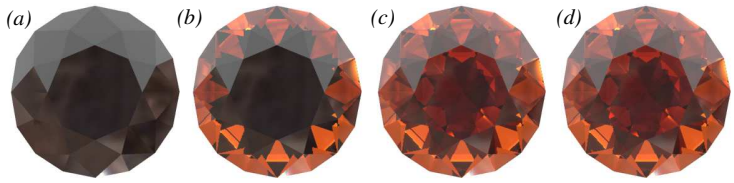
Figure 10: (*Bars charts, scale at right*:) Computation times in milliseconds for the construction of the facet tree only and for the full algorithm. (*Lines, scale at left:*) number of facets in the facet tree as a function of the number of internal reflections, (1) in the case where the facet tree is full down to the requested reflection level, and (2) when the facet tree is optimized for an error bound of 4%. Two different models were used: model 1 (130 vertices, 115 faces) and 2 (120 vertices, 93 faces). The picture row shows, for model 1, the raw and optimized facet trees as well as the corresponding images. Parameters correspond to violet/blue sapphire.

Computation times In Figure 10 we present the computation times (in *ms*) and the numbers of facets in the resulting facet trees for two different models. This experiment shows that cleverly pruning the facet tree makes the computation more efficient. Typical framerates for less complex models (40 to 80 facets), in a 800×600 window, range from 12 to 30 *fps* at depths 2 and 3.

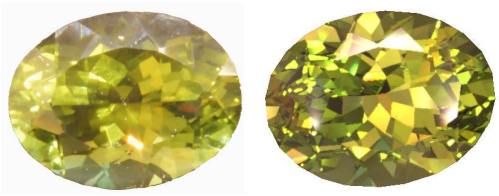
Comparison to real images On the top-right of Figure 11 one can see a comparison between a simulated tourmaline and a picture of a real one. The simulation was performed using the absorbance values and the geometric model displayed in Section 6.1, and captured lighting conditions.

Because it was impossible to precisely duplicate the geometry of the gems and the lighting conditions, and because the appearance of gems is very sensitive to both, one cannot expect a pixel-by-pixel correspondence between the images; one can, on the other hand, evaluate the phenomenological similarities and differences quite well. Both images display a change in color along the optical axis (oriented at approximately 60° in the image plane) hence the bluish tint on the bottom of each image (this effect, also seen on stones *(e),(f)* and *(g)* proves the need to account for birefringency in the simulation). Color ranges are quite similar, as well as flash effects (from a qualitative point of view) thanks to the HDR rendering.

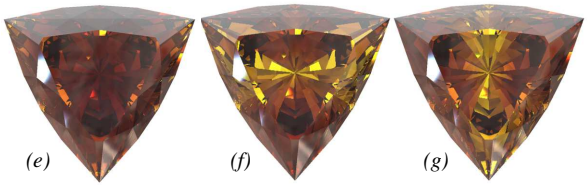
However one can complain that the luster of the stone (responsible for the irridescent colors on the top faces in the photography)



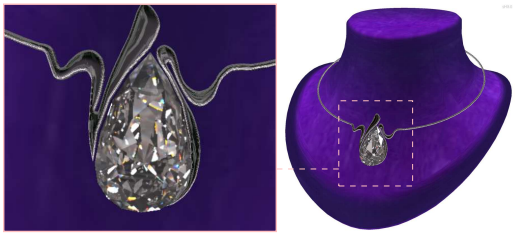
Grenat for different depths of the facet tree. *(a)* no internal reflection; *(b)* single internal reflection; *(c)* and *(d)*: 2 and 3 reflections; Note that after 2 reflections, very little is added to the image.



Left: real tourmaline, *right*: simulated stone with same model and similar lighting condition and viewpoint. See Section 6.2.



Andalusite. Optical axis *(e)* toward the eye,*(f)* updown and *(g)* left-to-right.



Example of application in Jewellery prototyping (diamond).

Figure 11: Various examples of results from our hardware rendering algorithm.

is not simulated, which would be a challenging problem to solve using graphics hardware.

7 Conclusion

We have presented a solution for rendering faceted colored gemstones based on the analysis of optical phenomena, their simplification and their adaptation to a hardware-based implementation. We have shown – we think for the first time – that it is possible to obtain some very visually pleasant and mostly physically correct images of faceted gemstones in real time by using a number of wise approximations of the real phenomena.

Our implementation benefits from the high level programming of today's graphic cards [Mark et al. 2003], which allows a comprehensive and portable implementation. Moreover, the rendering speed of our system offers new possibility for observing gemstones: not only the viewpoint can be changed in real time, but also the stone physical properties and geometry.

Our model can easily be extended to biaxial media. Other important features of gemstones, such as inclusions, color zoning and luster would be interesting to incorporate as well.

Acknowledgments

Many thanks to John Hughes for his careful reading of the paper, to Laurence Boissieux for the jewellery models and to the reviewers for helpful suggestions.

References

BEYERLE, G., AND MCDERMID, I. S. 1998. Ray tracing formulas for refraction and internal reflection in uniaxial crystals. *Applied Optics 37*, 34 (Dec.), 7947–7953.

BOGART, R., KAINZ, F., AND HESS, D. 2003. Openexr image file format. *Siggraph Technical Sketches and Applications* (July).

BORGES, C. F. 1991. Trichromatic approximation for computer graphics illumination models. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, ACM Press, Eurographics, 101–104.

BORN, M., AND WOLF, E. 1999. *Principles of Optics, Electromagnetic Theory of Propagation, Interference and Diffraction of Light*. Cambridge University Press.

C.MCCLAIN, S., HILLMAN, L. W., AND CHIPMAN, R. A. 1993. Polarization ray tracing in anisotropic optically active media. ii. theory and physics. *Journal of Optical Society of America 10*, 11 (Nov.).

DEBEVEC, P. 1998. Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, ACM Press, 189–198.

DOYLE, A. 2000. Screen gems: Cad/cam technologies are replacing traditional methods of jewely making. *Computer Graphics World* (July).

GLASSNER, A. S. 1995. *Principles of Digital Image Synthesis*. Morgan Kaufmann.

GREENE, N. 1986. Environment mapping and other applications of world projections. *IEEE Comput. Graph. Appl. 6*, 11, 21–29.

HECKBERT, P. S., AND HANRAHAN, P. 1984. Beam tracing polygonal objects. In *Computer Graphics (SIGGRAPH '84 Proceedings)*, H. Christiansen, Ed., vol. 18.

HUGHES, R. W. 1997. *Ruby & Sapphire*. RWH Publishing.

LINDHOLM, E., KILGARD, M. J., AND MORETON, H. 2001. A user-programmable vertex engine. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM Press, 149–158.

MARK, W. R., GLANVILLE, R. S., AKELEY, K., AND KILGARD, M. J. 2003. Cg: a system for programming graphics hardware in a c-like language. *ACM Trans. Graph. 22*, 3, 896–907.

NASSAU, K. 2001. *The Physics and Chemistry of Colour*. John Wiley & Sons.

NEIDER, J., DAVIS, T., AND WOO, M. 1993. *The OpenGL Programming Guide – OpenGL Version 1.2*. Addision-Wesley. Third Edition.

NVIDIA CORPORATION. 2000. Perfect reflections and specular lighting effects with cube environment mapping. Tech. rep. http://developer.nvidia.com/.

PEERCY, M. S. 1993. Linear color representations for full spectral rendering. In *Comp. Graphics (SIGGRAPH '93 Proceedings)*, J. T. Kajiya, Ed., vol. 27, 191–198.

SPENCER, G., SHIRLEY, P., ZIMMERMAN, K., AND GREENBERG, D. P. 1995. Physically-based glare effects for digital images. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, ACM Press.

SUN, Y., FRACCHIA, F. D., AND DREW, M. S. 1999. Rendering the phenomena of volume absorption in homogeneous transparent materials. In *2nd Annual IASTED International Conference on Computer Graphics and Imaging (CGIM'99)*, 283–288. http://fas.sfu.ca/pub/cs/mark/Cgim99/volumeAbs.ps.gz.

SUN, Y., FRACCHIA, F. D., AND DREW, M. S. 2000. Rendering diamonds. In *Proceedings of the 11th Western Computer Graphics Symposium (WCGS)*, 9–15.

SUN, Y., FRACCHIA, F. D., AND DREW, M. S. 2000. Rendering light dispersion with a composite spectral model. In *International Conference on Color in Graphics and Image Processing - CGIP'2000*.

TANNENBAUM, D. C., TANNENBAUM, P., AND WOZNY, M. J. 1994. Polarization and birefringency considerations in rendering. In *Comp. Graphics (SIGGRAPH '94 Proceedings)*, ACM Press, 221–222 (Extended version available on CD–ROM).

THOMAS, S. 1986. Dispersive refraction in ray tracing. *Visual Computer 2*, 3–8.

WILKIE, A., TOBLER, R. F., AND PURGATHOFER, W. 2000. Raytracing of dispersion effects in transparent materials. In *WSCG Conference Proceedings*. http://citeseer.nj.nec.com/wilkie00raytracing.html.

WILKIE, A., TOBLER, R. F., AND PURGATHOFER, W. 2001. Combined rendering of polarization and fluorescence effects. Tech. Rep. 186-2-01-11. Available at www.cg.tuwien.ac.at/research/TR/01/.

WOLF, E. 1959. Coherence properties of partially polarized electromagnetic radiation. *Il Nuovo Cimento 8*, 6 (september), 1165–1181.

WOLFF, L., AND KURLANDER, D. J. 1990. Ray tracing with polarization parameters. *IEEE Computer Graphics and Applications 10*, 6 (november/december), 44–55.

WYSZECKI, G., AND STILES, W. 1982. *Color science: Concepts and Methods. Quantitative Data and Formulas*. Wiley.

YOKOI, S., KURASHIGE, K., AND ICHIRO TORIWAKI, J. 1986. Rendering gems with asterism and chatoyancy. *The Visual Computer 2*, 5 (Sept.), 307–312.

YUAN, Y., KUNII, T. L., INAMATO, N., AND SUN, L. 1988. Gemstone fire: Adaptive dispersive ray tracing of polyhedrons. *The Visual Computer 4*, 5 (Nov.), 259–270.

Accurate Detection of Symmetries in 3D Shapes

AURÉLIEN MARTINET, CYRIL SOLER, NICOLAS HOLZSCHUCH and FRANÇOIS X. SILLION
ARTIS, INRIA Rhône-Alpes

We propose an automatic method for finding symmetries of 3D shapes, that is, isometric transforms which leave a shape globally unchanged. These symmetries are deterministically found through the use of an intermediate quantity: the generalized moments. By examining the extrema and spherical harmonic coefficients of these moments, we recover the parameters of the symmetries of the shape. The computation for large composite models is made efficient by using this information in an incremental algorithm capable of recovering the symmetries of a whole shape using the symmetries of its subparts. Applications of this work range from coherent remeshing of geometry with respect to the symmetries of a shape to geometric compression, intelligent mesh editing, and automatic instantiation.

Categories and Subject Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling—*Curve, surface, solid and object representations*

General Terms: Algorithms

1. INTRODUCTION

Many shapes and geometrical models exhibit *symmetries*: isometric transforms that leave the shape globally unchanged. Using symmetries, one can manipulate models more efficiently through coherent remeshing or intelligent mesh editing programs. Other potential applications include model compression, consistent texture-mapping, model completion, and automatic instantiation.

The symmetries of a model are sometimes made available by the creator of the model and represented explicitly in the file format the model is expressed in. Usually, however, this is not the case, and automatic translations between file formats commonly result in the loss of this information. For scanned models, symmetry information is also missing by nature.

In this article, we present an algorithm that automatically retrieves symmetries in a geometrical model. Our algorithm is independent of the tesselation of the model; in particular, it does not assume that the model has been tesselated in a manner consistent with the symmetries we attempt to identify, and it works well on noisy objects such as scanned models. Our algorithm uses a new tool, the *generalized moment* functions. Rather than computing these functions explicitly, we directly compute their spherical harmonic coefficients, using a fast and accurate technique. The extrema of these functions and their spherical harmonic coefficients enable us to deterministically recover the symmetries of a shape.

For composite shapes, that is, shapes built by assembling simpler structures, we optimize the computation by applying the first algorithm to the subparts, then iteratively building the set of symmetries of

Authors' addresses: A. Martinet, C. Soler, N. Holzschuch, F. X. Sillion, ARTIS, INRIA Rhône-Alpes, Saint Ismier, France; email: Aurelien. Martinet@imag.fr.

the composite shape, taking into account both the relative positions of the subparts and their relative orientations.

We envision many applications for our work, including geometric compression, consistent mesh editing, and automatic instantiation.

This article is organized as follows. In the following section, we review previous work on identifying geometric symmetries on 2D and 3D shapes. Then in Section 3, we present an overview of the symmetry-detection problem and the quantities used in our algorithms. In Section 4, we introduce the generalized moments and our method to compute them efficiently; in Section 5, we present our algorithm for identifying symmetries of a shape. The extension of this algorithm to composite shapes is then presented in Section 6., Finally, in Section 7, we show various applications of our algorithm.

2. RELATED WORK

Early approaches to symmetry detection focused on the 2D problem. Attalah [1985], Wolter et al. [1985] and Highnam [1985] present methods to reduce the 2D-symmetry detection problem to a 1D pattern matching problem for which efficient solution are known [Knuth et al. 1977]. Their algorithms efficiently detect all possible symmetries in a point set but are highly sensitive to noise.

Identifying symmetries for 3D models is much more complex, and little research on this subject has been published. Jiang and Bunke [1991] present a symmetry-detection method, restricted to rotational symmetry, based on a scheme called generate and test, first finding hypothetical symmetry axes, then verifying these assumptions. This method is based on a graph representation of a solid model and uses graph theory. The dependency between this graph representation and the mapping between points makes their method highly dependent on the topology of the mesh and sensitive to small modifications of the object geometry. Brass and Knauer [2004] provide a model for general 3D objects and give an algorithm to test congruence or symmetry for these objects. Their approach is capable of retrieving symmetry groups of an arbitrary shape but is also topology-dependent since it relies on a mapping between points of the model. Starting from an octree representation, Minovic et al. [1993] describe an algorithm based on octree traversal to identify symmetries of a 3D object. Their algorithm relies on PCA to find the candidate axis; PCA, however, fails to identify axes for a large class of objects, including highly symmetric objects such as regular solids.

All these methods try to find strict symmetries for 3D models. As a consequence, they are sensitive to noise and data imperfections. Zabrodsky et al. [1995] define a measure of symmetry for nonperfect models, defined as the minimum amount of work required to transform a shape into a symmetric shape. This method relies on the ability to first establish correspondence between points, a very restrictive precondition.

Sun and Sherrah [1997] use the *Extended Gaussian Image* to identify symmetries by looking at correlations in the Gaussian image. As in Minovic et al. [1993], they rely on PCA to identify potential axes of symmetry, thus possibly failing on highly symmetric objects. More recently, Kazhdan et al. [2004] introduced the *symmetry descriptors*, a collection of spherical functions that describe the measure of a model's rotational and reflective symmetry with respect to every axis passing through the center of mass. Their method provides good results in the shape identification but involves a surface integration for each sampled direction; this surface integration is carried on a voxel grid. Using the symmetry descriptors to identify symmetries requires an accurate sampling in all directions, making their algorithm very costly for an accurate set of results. In contrast, our algorithm only computes a deterministic small number of surface integrals, which are performed on the shape itself, and still provides very accurate results. Effective complexity comparisons will be given in Section 8.

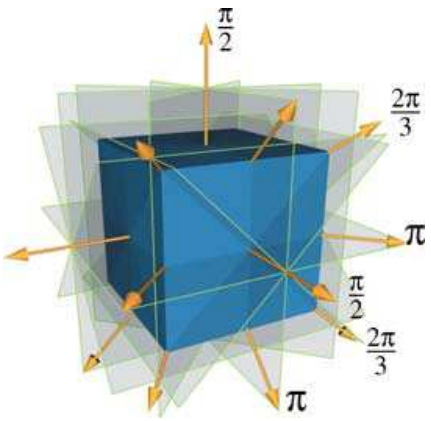


Fig. 1. Mirror symmetries and rotational symmetries found by our algorithm for a cube (for clarity, not all elements are represented).

3. OVERVIEW

Considering a surface $\mathcal{S}$, the *symmetries* of $\mathcal{S}$ are the isometric transforms which map $\mathcal{S}$ onto itself, in any coordinate system centered on its center of gravity. Symmetries of a shape form a group for the law of function composition with identity as its neutral element. For a given shape, the study of such a group relates to the domain of mathematical crystallography [Prince 2004].

The group of the cube, for instance, contains 48 elements (see Figure 1): the identity, eight 3−fold rotations around 4 possible axes, nine 4−fold rotations around 3 possible axes, six 2−fold rotations around 6 possible axes, nine mirror-symmetries, and fifteen other elements obtained by composing rotations and mirror symmetries.

Studying the group of isometries in $\mathbb{R}^3$ shows that, for a given isometry I, there always exists an orthonormal basis $(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z})$ into which the matrix of I takes the following form:

$$I(\lambda, \alpha) = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} \quad \text{with} \quad \begin{cases} \alpha \in [0, 2\pi[\\ \lambda = \pm 1 \end{cases}$$

As suggested by the example of the cube, this corresponds to 3 different classes of isometries: rotations, mirror symmetries, and their composition, depending whether λ is positive and/or $\alpha = 0 (\mathrm{mod}\ \pi)$. Finding a symmetry of a shape thus resolves into finding a vector $\boldsymbol{X}$ — which we call the *axis* of the isometry — and an angle α — which we call the *angle* of the isometry — such that $I(\lambda, \alpha)$ maps this shape onto itself.

However, finding all symmetries of a shape is much more difficult than simply checking whether a given transform actually is a symmetry. In particular, the naive approach that would consist of checking as many sampled values of $(\boldsymbol{X}, \lambda, \alpha)$ as possible to find a symmetry is far too costly. We thus need a deterministic method for finding good candidates.

Our approach to finding symmetries is to use intermediate functions, which set of symmetries is a superset of the set of symmetries of the shape itself, but for which computing the symmetries is much easier. By examining these functions, we will derive in Section 5 a deterministic algorithm which finds a finite number of possible candidates for $\boldsymbol{X}$, λ, and α. Because some unwanted triplets of values will appear during the process, these candidates are then checked back on the original shape. Choosing a family of functions which fulfill these requirements is easy. More difficult is the

task of finding such functions for which computing the symmetries can be done both accurately and efficiently.

Inspired by the work on principal component analysis [Minovic et al. 1993], we introduce the *generalized moment* functions of the shape for this purpose. These functions will be the topic of Section 4. These functions, indeed, have the same symmetries as the shape itself plus a small number of extra candidates. Furthermore, we propose an elegant framework based on spherical harmonics to accurately and efficiently find their symmetries.

A second contribution of this article is to extend the proposed algorithm into a constructive algorithm which separately computes the symmetries of subcomponents of an object—using the first method—, and then associates this information to compute symmetries of the whole composite shape. This constructive algorithm proves to be more accurate in some situations and more efficient when it is possible to decompose an object according to its symmetries. It is presented in Section 6.

4. GENERALIZED MOMENTS

In this section, we introduce a new class of functions: the generalized moments of a shape. We then show that these functions have at least the same symmetries as the shape itself and that their own symmetries can be computed in a very efficient way.

4.1 Definition

For a surface $\mathcal{S}$ in a 3-dimensional domain, we define its *generalized moment* of order $2p$ in direction $\boldsymbol{\omega}$ by

$$\mathcal{M}^{2p}(\boldsymbol{\omega}) = \int_{\boldsymbol{s}\in\mathcal{S}} \|\boldsymbol{s} \times \boldsymbol{\omega}\|^{2p} \, \mathrm{d}\boldsymbol{s}. \tag{1}$$

In this definition, $\boldsymbol{s}$ is a vector which links the center of gravity of the shape (placed at the origin) to a point on the surface, and $\mathrm{d}\boldsymbol{s}$ is thus an infinitesimal surface element. $\mathcal{M}^{2p}$ itself is a directional function.

It should be noted that, considering $\mathcal{S}$ to have some thickness dt, the expression $\mathcal{M}^2(\boldsymbol{\omega})dt$ (i.e., the generalized moment of order 2) corresponds to the moment of inertia of the thin shell $\mathcal{S}$ along $\boldsymbol{\omega}$, hence the name of these functions. Furthermore, the choice of an even exponent and a cross-product will lead to very interesting properties.

4.2 Shape Symmetries and Moments

Symmetry properties of a shape translate into symmetry properties of its moment functions. We now introduce a theorem that we will be rely on (see proof in Appendix):

THEOREM 1. *Any symmetry I of a shape $\mathcal{S}$ also is a symmetry of all its $\mathcal{M}^{2p}$ moment functions:*

$$I(\mathcal{S}) = \mathcal{S} \quad \Rightarrow \quad \forall \boldsymbol{\omega} \;\; \mathcal{M}^{2p}(I(\boldsymbol{\omega})) = \mathcal{M}^{2p}(\boldsymbol{\omega}).$$

Furthemore, if $\mathcal{M}^{2p}$ has a symmetry I with axis $\boldsymbol{\omega}$, then the gradient of $\mathcal{M}^{2p}$ is null at $\boldsymbol{\omega}$:

$$\forall \boldsymbol{\omega} \quad \mathcal{M}^{2p}(I(\boldsymbol{\omega})) = \mathcal{M}^{2p}(\boldsymbol{\omega}) \quad \Rightarrow \quad (\boldsymbol{\nabla}\mathcal{M}^{2p})(\boldsymbol{\omega}) = 0.$$

This theorem implies that the axes of the symmetries of a shape are to be found in the intersection of the sets of directions which zero the gradients of each of its moment functions. The properties are not reciprocal, however. Once the directions of the zeros of the gradients of the moment functions have been found, they must be checked on the shape itself to eliminate false positives.

4.3 Efficient Computation

At first sight, looking for the zeros of the gradient of the moment functions requires precise and dense sampling of these functions which would be very costly using their integral form of Equation (1). We thus present an efficient method to compute the generalized even moment functions of a shape, using spherical harmonics. In particular, we can accurately compute the spherical harmonic coefficients of the moment functions without sampling these functions. The search for zeros in the gradient will then be performed efficiently on the spherical harmonic decomposition itself.

Spherical Harmonics. We use real-valued spherical harmonics [Hobson 1931] to represent directional functions. Real spherical harmonics are defined, for integers $l \geq 0$ and $-l \leq m \leq l$, by:

$$Y_l^m(\theta,\varphi) = \begin{cases} \sqrt{2}\, N_l^m P_l^m(cos\theta)\cos(m\varphi) & \text{for } 0 < m \leq l \\ N_l^m P_l^0\ (cos\theta) & \text{for } m = 0 \\ \sqrt{2}\, N_l^m P_l^{-m}(cos\theta)\sin(m\varphi) & \text{for } -l \leq m < 0 \end{cases}$$

where P_l^m are the associated Legendre polynomials; the normalization constants N_l^m are such that the spherical harmonics form an orthonormal set of functions for the scalar product:

$$< f, g >= \int_{\|\omega\|=1} f(\omega) g(\omega)\, \mathrm{d}\omega.$$

This corresponds to choosing:

$$N_l^m = \sqrt{\frac{2l+1}{4\pi}\frac{(l-|m|)!}{(l+|m|)!}}.$$

We will use the following very powerful property of spherical harmonics. Any spherical harmonic of degree l can be expressed in a rotated coordinate system using harmonics of same degree and coefficients depending on the rotation R:

$$Y_l^m \circ R = \sum_{-l \leq m' \leq l} D_l^{m,m'}(R) Y_l^{m'}. \tag{2}$$

Any combination of spherical harmonics of degree less than l can therefore be expressed in a rotated coordinate system, using spherical harmonics of degree less than l, without loss of information. Coefficients $D_l^{m,m'}(R)$ can efficiently be obtained using recurrence formula [Ivanic and Ruedenberg 1996] or directly computed [Ramamoorthi and Hanrahan 2004].

Computation of Moment Functions. As defined by Equation (1), the $2p-$moment function of a shape $\mathcal{S}$ is expressed as:

$$\begin{aligned} \mathcal{M}^{2p}(\omega) &= \int_{\boldsymbol{s}\in\mathcal{S}} \|\boldsymbol{s}\times\omega\|^{2p}\, \mathrm{d}\boldsymbol{s} \\ &= \int_{\boldsymbol{s}\in\mathcal{S}} \|\boldsymbol{s}\|^{2p} \sin^{2p}\beta\, \mathrm{d}\boldsymbol{s} \end{aligned}$$

In this expression, β is the angle between s and ω.

Function $\beta \mapsto \sin^k \beta$ has angular dependence on β only and therefore decomposes into zonal harmonics (i.e., harmonics Y_l^m for which $m = 0$). Performing the calculation shows that, when k is even, the decomposition is finite. Setting $k = 2p$, we obtain :

$$\sin^{2p}\beta = \sum_{l=0}^{p} S_p^l Y_{2l}^0(\beta, .)$$

with:

$$S_p^l = \frac{\sqrt{(4l+1)\pi}}{2^{2l}} \sum_{k=l}^{2l} (-1)^k \frac{2^{2p+1} p!(2k)!(p+k-l)!}{(2(p+k-l)+1)!(k-l)!k!(2l-k)!}. \tag{3}$$

For the sake of completeness, we provide the corresponding derivation and the proof of the finite decomposition in the appendix section of this article.

Let R_s be a rotation which maps z, unit vector along $z-$axis, to s. Using Equation (2) for rotating the Y_{2l}^0 zonal harmonics, we have :

$$\sin^{2p}\beta = \sum_{l=0}^{p} S_p^l \sum_{m=-2l}^{2l} D_{2l}^{0,m}(R_s) Y_{2l}^m(\omega).$$

And finally:

$$\mathcal{M}^{2p}(\omega) = \sum_{l=0}^{p} \sum_{m=-2l}^{2l} C_{2l,m}^{2p} Y_{2l}^m(\omega), \tag{4}$$

using

$$C_{2l,m}^{2p} = S_p^l \int_{s\in\mathcal{S}} \|s\|^{2p} D_{2l}^{0,m}(R_s)\, \mathrm{d}s. \tag{5}$$

Equation (4) says that $\mathcal{M}^{2p}$ decomposes into a finite number of spherical harmonics, and Equation (5) allows us to directly compute the coefficients. The cost of computing $\mathcal{M}^{2p}$ is therefore $(p+1)(2p+1)$ surface integrals (one integral per even order of harmonic, up to order $2p$). This is much cheaper than the alternative method of computing the scalar product of $\mathcal{M}^{2p}$ as defined by Equation (1) with each spherical harmonic basis function: this would indeed require many evaluations of $\mathcal{M}^{2p}$, which itself is defined as a surface integral. Furthermore, numerical accuracy is only a concern when computing the $C_{2k,p}^m$ coefficients, and we can now compute both $\mathcal{M}^{2p}$ and its gradient analytically from Equation (4).

5. FINDING SYMMETRIES OF A SINGLE SHAPE

In this section, we present our algorithm for identifying symmetries of a shape seen as a single entity as opposed to the algorithm presented in the next section where the shape is considered as an aggregation of multiple subparts. For a given shape, we want to determine the axis $\boldsymbol{X}$ and the (λ, α) parameters of the potential isometries, using the generalized moment functions, and check the isometries found against the actual shape.

Central symmetries ($\lambda = -1$ and $\alpha = \pi$) form a specific case since, by construction, $\mathcal{M}^{2p}$ always has a central symmetry. Because central symmetries also do not require an axis, we treat this case directly while checking the other candidate symmetries on the shape itself in Section 5.3.

5.1 Determination of the Axis

As we saw in Section 4.2, the axis of isometries which let a shape globally unchanged also zero the gradient of the generalized even moments of this shape. We thus obtain a superset of them by solving for:

$$\nabla(\mathcal{M}^{2p})(\omega) = \mathbf{0}.$$

In a first step, we estimate a number of vectors which are close to the actual solutions by refining the sphere of directions starting from an icosahedron. In each face, the value of $\|\nabla(\mathcal{M}^{2p})(\omega)\|^2$ is examined

in several directions, and faces are sorted by order of the minimal value found. Only faces with small minimum values are refined recursively. The number of points to look at in each face, as well as the number of faces to keep at each depth level, are constant parameters of the algorithm.

In a second step, we perform a steepest descent minimization on $\|\nabla(\mathcal{M}^{2p})(\omega)\|^2$, starting from each of the candidates found during the first step. For this we need to evaluate the derivatives of $\|\nabla(\mathcal{M}^{2p})\|$ which we do using analytically computed second-order derivatives of the spherical harmonics along with Equation (4). The minimization converges in a few steps because starting positions are by nature very close to actual minima. This method has the double advantage that (1) the derivatives are very efficiently computed and (2) no approximation is contained into the calculation of the direction of the axis beyond the precision of the calculation of the $C^{2p}_{2l,m}$ coefficients.

During this process, multiple instances of the same direction can be found. We filter them out by estimating their relative distance. While nothing in theory prevents the first step from missing the area of attraction of a minimum, it works very well in the present context. Indeed, moment functions are very smooth, and shapes having two isometries with very close—yet different—axis are not common.

Finally, because all moment functions, whatever their order, must have an extremum in the direction of the axis of the symmetries of the shape, we compute such sets of directions for multiple moment functions (e.g., $\mathcal{M}^4$, $\mathcal{M}^6$ and $\mathcal{M}^8$) but keep only those which simultaneously zero the gradient of all these functions, which in practice leaves none or very few false positives to check for.

5.2 Determination of Rotation Parameters

After finding the zero directions for the gradient of the moment functions, we still need to find the parameters of the corresponding isometric transforms. This is done deterministically by studying the spherical harmonic coefficients of the moment functions themselves. We use the following properties.

PROPERTY 1. *A function has a mirror symmetry S_z around the $z = 0$ plane if and only if all its spherical harmonic coefficients for which $l + m$ is even are zero (i.e., it decomposes onto $z-$symmetric harmonics only). In the specific case of the moment functions:*

$$\forall\omega \;\; \mathcal{M}^{2p}(\omega) = \mathcal{M}^{2p}(S_z\omega) \;\; \Leftrightarrow \;\; m \equiv 0(mod\ 2) \Rightarrow C^{2p}_{2l,m} = 0.$$

PROPERTY 2. *A function has a revolution symmetry around the z axis if and only if it decomposes onto zonal harmonics only, that is,*

$$\forall l \;\; \forall m \quad m \neq 0 \;\; \Rightarrow \;\; C^m_l = 0.$$

PROPERTY 3. *A function is self-similar through a rotation R_α of angle α around z if and only if all its spherical harmonic coefficients C^m_l verify:*

$$\forall l \;\; \forall m \quad C^m_l = \cos(m\alpha)C^m_l - \sin(m\alpha)C^{-m}_l. \tag{6}$$

Property 3 can be adapted to check if the function is self-similar through the composition of a rotation and a symmetry with the same axis (i.e., the case $\lambda = -1$ as defined in Section 3). In this case, the equation to be checked for is:

$$\forall l \;\; \forall m \quad (-1)^{l+m}C^m_l = \cos(m\alpha)C^m_l - \sin(m\alpha)C^{-m}_l. \tag{7}$$

These properties are easily derived from the very expression of the spherical harmonic functions [Hobson 1931].

Before using these properties, the moment function must be expressed in a coordinate system where the z axis coincides with the previously found candidate axis. This is performed using the rotation formula in Equation (2). Then checking for Properties 1 and 2 is trivial provided that some tolerance is accepted on the equalities. Using Property 3 is more subtle; coefficients of the function are first examined by order of decreasing m. For $\lambda = 1$, for instance, when the first nonzero value of C_l^m is found, Equation (6) is solved by:

$$\tan\frac{m\alpha}{2} = \frac{C_l^{-m}}{C_l^m}, \quad \text{that is,} \quad \alpha = \frac{2}{m}\arctan\left(\frac{C_l^{-m}}{C_l^m}\right) + \frac{k\pi}{m},$$

then all the remaining coefficients are checked with the obtained values of α. If the test passes, then α is the angle of an existing rotation symmetry for the moment function. A very similar process is used to search for α when $\lambda = -1$.

The error tolerance used when checking for Properties 1, 2, and 3 can be considered as a way of detecting approximate symmetries on objects. We will show in the results section that symmetries can indeed be detected on noisy data such as scanned models.

5.3 Filtering Results

The condition extracted from Theroem 1 is a necessary condition only. To avoid false positives, the directions and rotation angles obtained from the moment functions must therefore be verified on the shape itself. We do this using a *symmetry measure* inspired by the work of Zabrodsky et al. [1995]. Let $\mathcal{S}$ and $\mathcal{R}$ be two tessellated shapes. Let $V_\mathcal{S}$ and $V_\mathcal{R}$ be the mesh vertices of $\mathcal{S}$ and $\mathcal{R}$. We define the *measure* d_M between $\mathcal{S}$ and $\mathcal{R}$ by:

$$d_M(\mathcal{S},\mathcal{R}) = \max_{p\in V_\mathcal{S}}(\min_{q\in\mathcal{R}} \|p-q\|). \tag{8}$$

The *symmetric measure* $d_A(\mathcal{S})$ of a shape $\mathcal{S}$ with respect to a symmetry A is then defined by:

$$d_A(\mathcal{S}) = \max(d_M(\mathcal{S}, A\mathcal{S}), d_M(A\mathcal{S}, \mathcal{S})).$$

It should be noted that this definition is different from that of the *Hausdorff distance* since, in Equation (8), not all points of $\mathcal{S}$ are considered but only the mesh vertices, whereas all points of $\mathcal{R}$ are used. However, because $\mathcal{S}$ is polyhedral, $d_A(\mathcal{S}) = 0$ still implies that $A\mathcal{S} = \mathcal{S}$.

Computing d_A is costly, but fortunately we only compute it for a few choices of A which are the candidates we found at the previous step of the algorithm. This computation is much cheaper than computing a full symmetry descriptor [Kazhdan et al. 2004] for a sufficient number of directions to reach the precision of our symmetry detection algorithm.

5.4 Results

Complete Example. The whole process is illustrated in Figure 2. Starting from the original object (a), the moment functions of orders 4, 6, and 8 are computed (see, e.g., $\mathcal{M}^8$ in (b)). The gradients of these moments are then computed analytically (c) and used for finding the directions of the minima. The unfiltered set of directions contains 7 directions among which only 3 are common extrema of $\mathcal{M}^4$, $\mathcal{M}^6$, and $\mathcal{M}^8$. This set of 3 directions ($\boldsymbol{D}_1$,$\boldsymbol{D}_2$, and $\boldsymbol{D}_3$) must contain the axes of the symmetries of the shape. After checking the symmetry axis and parameters on the actual shape, $\boldsymbol{D}_1$ is revealed as the axis of a 2-fold symmetry which is the composition of the two remaining mirror symmetries of axes $\boldsymbol{D}_2$ and $\boldsymbol{D}_3$.

The example of the cube, shown in Figure 1, illustrates the extraction of rotations and mirror symmetries. Experiments have shown that our method finds all 48 symmetries whatever the coordinate system the cube is expressed in originaly.

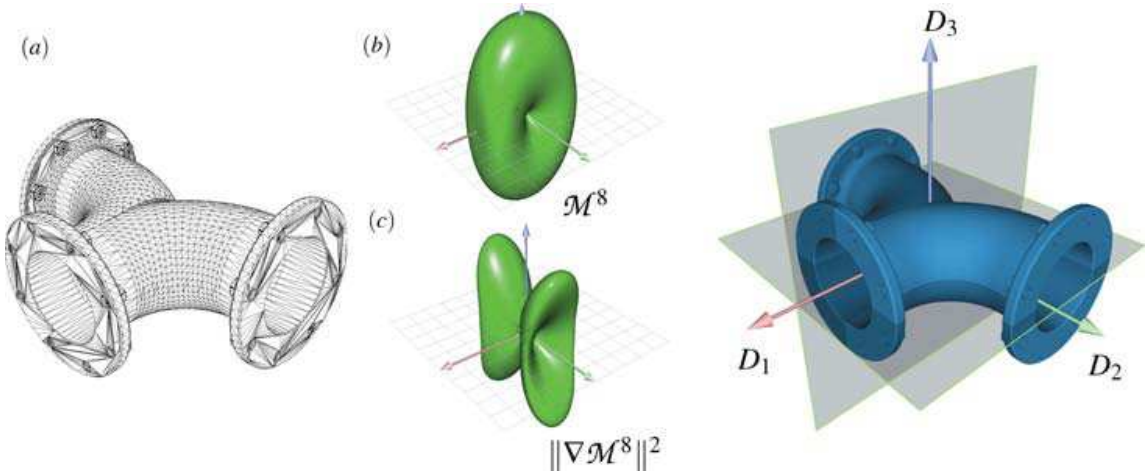


Fig. 2. Extraction of symmetries for a single shape. Starting from the original shape (a), generalized moments (b) and their gradients (c) are computed. The set of their common extrema directions contains the axes of the symmetries of the shape, depicted at right. Here, both mirror symmetries have been found as well as the 2-fold rotational symmetry. Note that the original shape is neither convex nor star-shaped and that the mesh is not consistent with the symmetries of the geometry.

Fig. 3. View of the three 3D models used in the robustness tests presented in Figure 4 shown with their symmetries. For the sake of clarity, we chose models with only one symmetry each.

Robustness Tests. We now study the sensitivity of our method to small perturbations of the 3D model in two different ways.

(1) *Noise.* We randomly perturb each vertex of each polygon independently in the original model by a fraction of the longest length of the model's bounding box.

(2) *Delete.* We randomly delete a small number of polygons in the model.

We use a set of three models to test the robustness of our method. These model as well as their symmetry are shown in Figure 3. For the sake of clarity, we use objects with only one symmetry.

In order to test the robustness of the method, we progressively increase the magnitude of the noise and let the algorithm automatically detect the symmetry. In our robustness tests, we consider shapes as single entities and use the first algorithm presented in Section 5 to detect these symmetries. To evaluate

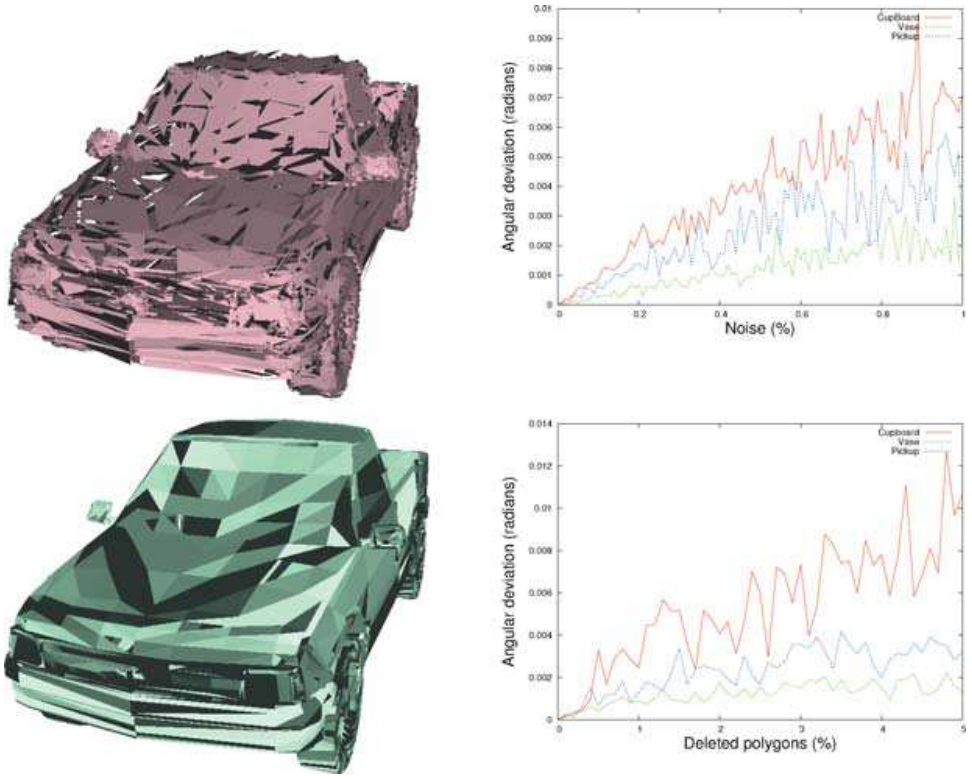


Fig. 4. We test the sensitivity of the method to noise by progressively increasing noise magnitude and letting the algorithm detect the symmetry for each of our three test models. We evaluate the accuracy of the results by computing the angular deviation between the axis found and the axis of the symmetry of the original model. *Top row:* We perturb each vertex of each polygon independently by a fraction of the longest length of the bounding box on each of the three test models. The left figure shows a noisy pick-up model with a noise magnitude of 1% and the right figure shows angular deviation evolution for the three models for a magnitude ranging from 0% to 1%. *Bottom row:* We randomly delete polygons of the models. The left figure shows a noisy pick-up obtained by deleting 5% of the polygons and the right figure shows angular deviation evolution by deleting 0% to 5% of the polygons of the three models. As can be seen from the curve, for small variations of the models, our method has approximatively linear dependency regarding noise and delivers high-quality results even for nonperfect symmetries.

the reliability of the results, we compute the angular deviation between the found axis of symmety and the real one, that is, computed with no noise. In our experiments, noise magnitude varies from 0 to 1% of the longest length of the model's bounding box, and the number of deleted polygons ranges from 0 to 5% of the total number of polygons in the model (see Figure 4).

The results of these experiments show that, for small variations, our method has approximatively linear dependency regarding noise and delivers high-quality results even for nonperfect symmetries. These statistical results can also be used to derive an upper bound on the mean angular error obtained as a function of the noise in the model.

5.4.1 *Application to Scanned Models.* We present in Figure 5 examples of applying the single-shape algorithm to scanned models, retreived from a Web database and used as is (see `http://shapes.aim-at-shape.net`). Our algorithm perfectly detects all the parameters of candidate symmetries for all these

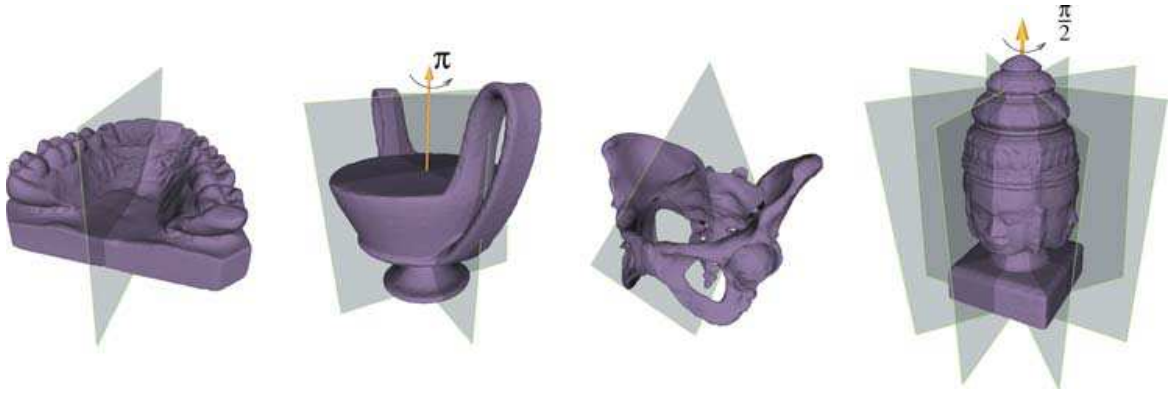


Fig. 5. Our algorithm perfectly detects approximate symmetries of scanned models. Detecting these symmetries requires relaxing the constraints when checking candidate symmetries on the model. Please note that these scanned models are by nature neither axis-aligned nor tesselated according to their symmetries. This illustrates the fact that our algorithm does not depend on the coordinate system nor on the mesh of the objects.

Table I. Computation times (in seconds) for the Four Scanned Models Presented in Figure 5

Model	Teeth	Vase	Pelvis	Angkor statue
# polygons	233, 204	76, 334	50, 000	163, 054
Computing moments*	33.7	11.8	7.26	23.26
Finding parameters	0.4	0.6	0.4	0.7
Checking candidates	9.4	11.1	5	12.2
Total	43.5	23.5	12.66	36.16

*Global computation times for moments of order 2 to 8

shapes. When testing these symmetries, one should allow a large enough symmetry distance error (as defined in Section 5.3) because these models are by nature not perfectly symmetric.

5.5 Discussion

Because the $\mathcal{M}^{2p}$ functions are trigonometric polynomials on the sphere, they have a maximum number of strict extrema depending on p: the larger p is, the more $\mathcal{M}^{2p}$ is able to capture the information of a symmetry, that is, to have an extremum in the direction of its axis. But because all moment functions *must* have a null gradient in this direction (according to Theorem 1), these extrema are bound to become nonstrict extrema for small values of p, and $\mathcal{M}^{2p}$ is forced to be constant on a subdomain of nonnull dimension. Using the cube as an example in which case $\mathcal{M}^2$ is a constant function a trigonometric polynomial of order 2 can simply not have enough strict extrema to represent all 12 distinct directions of the symmetries of the cube.

In all the tests we conducted, however, using moments up to order 10 has never skipped any symmetry on any model. But it would still be interesting to know the exact maximum number of directions permitted by moments of a given order.

6. FINDING SYMMETRIES OF GROUPS OF OBJECTS

In Section 5, we have presented an algorithm for finding the symmetries of single shapes. In this section, we present a *constructive* algorithm which recovers the symmetries of a group of objects—which we call *tiles* to indicate that together they form a larger object—from the symmetries and positions of each separate tile.

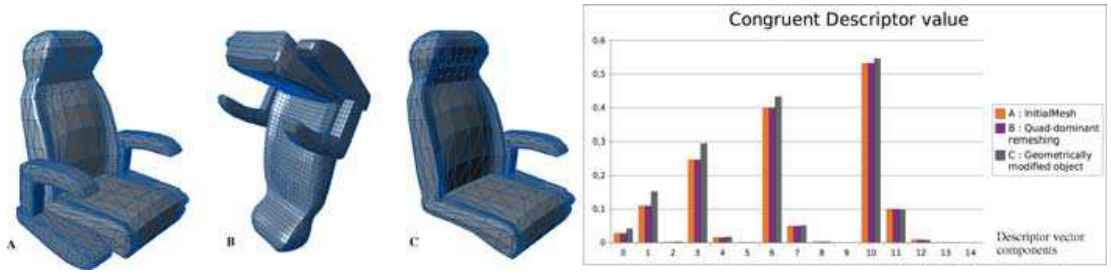


Fig. 6. This figure illustrates the reliability of our congruency descriptor (as defined by Equation (9)). Two identical objects meshed differently and expressed in two different coordinate systems (A and B) have extremely close descriptor vectors, but a slightly different object (C) has a different descriptor. The graphics on the right shows each component of the three descriptors.

The constructive algorithm first computes (if necessary) the symmetries of all separate tiles using the single shape algorithm. Then it detects which tiles are similar up to an isometric transform and finds the transformations between similar tiles. Then it explores all one-to-one mappings between tiles, discarding mappings which do not correspond to a symmetry of the group of tiles as a whole.

Section 6.2 explains how we detect similar tiles and Section 6.3 details the algorithm which both explores tile-to-tile mappings and finds the associated symmetry for the whole set of tiles.

Because it is always possible to apply the algorithm presented in Section 5 to the group of tiles, considering it as a single complex shape, questioning the usefulness of the constructive method is legitimate. For this reason, we will explain in Section 6.5 in which situations the constructive method is preferable to the algorithm for single shapes; but let us first explain the method itself.

6.1 Computing the Symmetries of Each Tile

If not available, the symmetries of each tile are computed using the algorithm presented in Section 5. When assembling known objects together, the economy of this computation can, of course, be performed by simply computing the symmetries of one instance for each class of different tiles.

6.2 Detecting Tiles Congruency

In this subsection, we introduce a shape descriptor suitable for detecting whether two shapes are identical up to an—unknown—isometry. We will use this tool for classifying tiles before trying to find a mapping of a composite object onto itself.

Let $\mathcal{S}$ be a shape and $C^{2p}_{2l,m}$ the spherical harmonic coefficients of its generalized even moment functions $\mathcal{M}^{2p}$ up to an order p. Our shape descriptor is defined as the $p(p+1)/2$-vector obtained by packing together the frequency energy of the spherical harmonic decomposition of all moments of $\mathcal{S}$ up to order p:

$$D_{2p} = \left[d^0_0, d^2_0, d^2_2, \ldots, d^{2p}_0, d^{2p}_2 \ldots d^{2p}_{2p}\right] \tag{9}$$

with

$$d^{2k}_{2l} = \sum_{-2l \le m \le 2l} \left(C^{2k}_{2l,m}\right)^2 \tag{10}$$

(See Figure 6). It has been shown by Kazhdan et al. [2003] that d^k_l, as defined in Equation (10), does not depend on the coordinate system the spherical harmonic decomposition is expressed in. This means that each d^{2p}_{2l}, and therefore D_{2p} itself, is not modified by isometric transforms of the shape. Mirror

Table II. Percentage of Tiles Matched by our Shape Descriptor That Are Effectively Identical For Our Test Scenes

Max order	39,557 Polygons 851 Tiles	182,224 Polygons 480 Tiles	515,977 Polygons 5,700 Tiles
2	92.1%	43.9%	92.3%
4	100%	78.0%	100%
6	100%	92.2%	100%
8	100%	100%	100%

Fig. 7. Scenes used for testing the object congruency descriptor. In each scene, the descriptor has been used to detect objects with similar geometry (but possibly different meshes) up to a rigid transform. Objects found to be congruent are displayed with the same color.

symmetries do not affect d_{2l}^{2p} either since they only change the sign of the coefficient for some harmonics in a coordinate system aligned with the axis.

Two tiles A and B are considered to be similar up to an isometric transform, at a precision ε, when:

$$\|D_{2p}(A) - D_{2p}(B)\| < \varepsilon.$$

Theoretically, this shape descriptor can produce false positives, that is, tiles that are not congruent but have the same descriptor, but it can not produce false negatives because of its deterministic nature. Our experiments have shown that using moments up to order 6 produces a sufficiently discriminant shape descriptor on all test scenes. This is illustrated in Table II where we present the average precision value, that is, the percentage of matched tiles that are actually identical up to an isometric transform, for a set of architectural scenes (Figure 7).

By definition, congruent tiles should have the same set of symmetries, possibly expressed in different coordinate systems. Since we know the symmetries of each of the tiles, we introduce this constraint, thereby increasing the discriminating power of our shape descriptor as shown in Table III.

6.3 Algorithm for Assembled Objects

6.3.1 *Overview.* Once we have determined all classes of congruent tiles, the algorithm examines all the one-to-one mappings of the set of all tiles onto itself which map each tile onto a similar tile. For each one-to-one mapping found, it determines the isometric transforms which are simultaneously compatible with each tile and its symmetries.

The algorithm works recursively: at the beginning of each recursion step, we have extracted two subsets of tiles, $\mathcal{H}_1$ and $\mathcal{H}_2$, of the composite shape $\mathcal{S}$, and we have computed the set of all possible isometric transforms that globally transform $\mathcal{H}_1$ into $\mathcal{H}_2$. Then, taking two new similar tiles, $\mathcal{S}_1 \in \mathcal{S} \setminus \mathcal{H}_1$

Table III. Percentage of Tiles Matched By Our Shape Descriptor That Are Effectively Identical Using the Added Constraint That Identical Tiles Must Have the Same Set of Symmetries Up to a Rigid Transform

Max order	39,557 Polygons 851 Tiles	182,224 Polygons 480 Tiles	515,977 Polygons 5,700 Tiles
2	95.6%	73.4%	97%
4	100%	96.0%	100%
6	100%	100%	100%
8	100%	100%	100%

and $\mathcal{S}_2 \in \mathcal{S} \setminus \mathcal{H}_2$, we restrict the set of isometric transforms to the isometric transforms that also map $\mathcal{S}_1$ onto $\mathcal{S}_2$ (but not necessarily $\mathcal{S}_2$ onto $\mathcal{S}_1$). Because these tiles have symmetries, this usually leaves multiple possibilities.

Note that the global symmetries found must always be applied with respect to the center of mass $\boldsymbol{g}$ of $\mathcal{S}$, according to the definition of a symmetry of $\mathcal{S}$.

At the end of the recursion step, we have the set of isometric transforms that map $\mathcal{H}_1 \cup \{\mathcal{S}_1\}$ onto $\mathcal{H}_2 \cup \{\mathcal{S}_2\}$.

Each recursion step narrows the choice of symmetries for $\mathcal{S}$. The recursion stops when either this set is reduced to identity transform or when we have used all the component tiles in the model. In the latter case, the isometric transforms found are the symmetries of the composite shape. The recursion is initiated by taking for $\mathcal{H}_1$ and $\mathcal{H}_2$ two similar tiles, that is, two tiles of the same class.

In the following paragraphs, we review the individual steps of the algorithm: finding all the isometric transforms which map tile $\mathcal{S}_1$ onto similar tile $\mathcal{S}_2$ and reducing the set of compatible symmetries of $\mathcal{S}$. We then illustrate the algorithm in a step-by-step example.

6.3.2 *Finding All the Isometries Which Transform a Tile onto a Similar Tile.* At each step of our algorithm, we examine pairs of similar tiles, $\mathcal{S}_1$ and $\mathcal{S}_2$, and we have to find all the isometries which map $\mathcal{S}_1$ onto $\mathcal{S}_2$.

If $\boldsymbol{g}_i$ is the center of mass of tile $\mathcal{S}_i$ and $\boldsymbol{g}$ is the center of mass of the composite shape $\mathcal{S}$, this condition implies that the isometries we are looking for transform vector $\boldsymbol{g}_1 - \boldsymbol{g}$ into $\boldsymbol{g}_2 - \boldsymbol{g}$. In order to generate the set of all isometric transforms that map $\mathcal{S}_1$ onto $\mathcal{S}_2$, we use the following property.

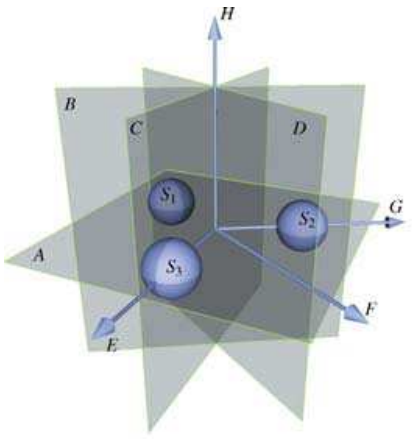
PROPERTY 4. *If J is an isometry that maps $\mathcal{S}_1$ onto a similar tile $\mathcal{S}_2$, then all the isometries K which map $\mathcal{S}_1$ onto $\mathcal{S}_2$ are of the following form:*

$$K = JT^{-1}AT \quad \textit{with} \quad A \in G_{\mathcal{S}_1} \quad \textit{such that} \quad A(\boldsymbol{g}_1 - \boldsymbol{g}) = \boldsymbol{g}_1 - \boldsymbol{g}, \tag{11}$$

where $G_{\mathcal{S}_1}$ is the group of symmetries of $\mathcal{S}_1$, and T is the translation of vector $\boldsymbol{g} - \boldsymbol{g}_1$ (refer to the Appendix for proof of this property).

This property states that, once we know a single *seed* isometric transform which maps $\mathcal{S}_1$ onto $\mathcal{S}_2$, we can generate all such transforms by using the elements of $G_{\mathcal{S}_1}$ in Equation (11).

6.3.3 *Finding a Seed Transform.* We need to find a seed transform J that maps $\mathcal{S}_1$ onto $\mathcal{S}_2$. For each tile, we extract a minimum set of independent vectors that correspond to extremas of their generalized even moment functions. The number of vectors needed depends on the symmetries of the tile. J is then defined as any isometric transform that maps the first set of vectors onto the second as well as vector $\boldsymbol{g}_1 - \boldsymbol{g}$ onto $\boldsymbol{g}_2 - \boldsymbol{g}$. Most of the time, a single isometric transform is possible at most. When multiple choices exist, the candidate transforms are checked onto the shapes using the distance presented in Section 5.3. This ensures that we find at least one seed transform.



Mapping	Symmetry
$S_1 \to S_1$ $S_2 \to S_2$ $S_3 \to S_3$	mirror-symmetry A
$S_1 \to S_3$ $S_3 \to S_1$ $S_2 \to S_2$	mirror-symmetry B 2-fold symmetry G
$S_1 \to S_2$ $S_2 \to S_1$ $S_3 \to S_3$	mirror-symmetry C 2-fold symmetry E
$S_2 \to S_3$ $S_3 \to S_2$ $S_1 \to S_1$	mirror-symmetry D 2-fold symmetry F
$S_1 \to S_2$ $S_2 \to S_3$ $S_3 \to S_1$	3-fold symmetry H

Fig. 8. Three spheres uniformly distributed on a circle in the z-plane. Etablishing all one-to-one mappings of the set of all tiles onto itself, which map each tile onto a similar tile, are used to detect all the symmetries of the shape. Note that the 3−fold symmetry H is detected and is associated to a circular permutation mapping.

6.3.4 *Ensuring Compatibility with Previous Isometries.* During the recursion, we need to store the current set of compatible isometries we have found. We do this by storing a minimal set of linearly independent vectors along with their expected images by these isometries. For example, if we have to store a symmetry of revolution, we store only one vector, the axis of the symmetry, and its image (itself). For mirror symmetries, rotations, and central symmetries, we store three independent vectors, along with their images by this isometric transform. For instance, in the case of a rotation of angle π around axis $\boldsymbol{X}$, we have:

$$\boldsymbol{X} \mapsto \boldsymbol{X} \quad \boldsymbol{Y} \mapsto -\boldsymbol{Y} \quad \boldsymbol{Z} \mapsto -\boldsymbol{Z}. \tag{12}$$

By examining all the one-to-one mappings of the set of all tiles onto itself, which map each tile onto a similar tile, we are able to detect all symmetries of the set of tiles (see Figure 8). Note in this example that the 3−fold symmetry H is detected and is associated to a circular permutation mapping.

6.4 Step-By-Step Example

Figure 9 presents a very simple example of a shape (a pair of pliers) composed of 3 tiles, $\mathcal{S}_1, \mathcal{S}_2$ (the handles), and $\mathcal{R}$ (the head). Two of the tiles are similar up to an isometric transform, $\mathcal{S}_1$ and $\mathcal{S}_2$. Figure 9 also displays the centers of mass, $\boldsymbol{g}_1$, and $\boldsymbol{g}_2$ of tiles $\mathcal{S}_1$ and $\mathcal{S}_2$ (which are not in the plane $z = 0$), and the center of mass $\boldsymbol{g}$ of the whole shape. In the coordinate systems centered on their respective centers of mass, $\mathcal{S}_1$ and $\mathcal{S}_2$ have a mirror symmetry of axis $\boldsymbol{Z}$, and $\mathcal{R}$ has a rotation symmetry around axis $\boldsymbol{X}$ of angle π.

Our constructive algorithm starts by selecting tile $\mathcal{R}$ and a similar tile (here, the only possible choice is $\mathcal{R}$).

Step 1. The algorithm explores the possibilities to transform $\mathcal{R}$ into itself. Two possibilities exist (a) the identity transform, and (b) the rotation around $\boldsymbol{X}$ of angle π, deduced from (a) by Property 4.
At this point, the algorithm branches, and either tries to map $\mathcal{S}_1$ to itself (branch 1) or to $\mathcal{S}_2$ (branch 2).
Branch 1, Step 1. The algorithm tries to match $\mathcal{S}_1$ to itself. The only compatible transform is the identity transform.

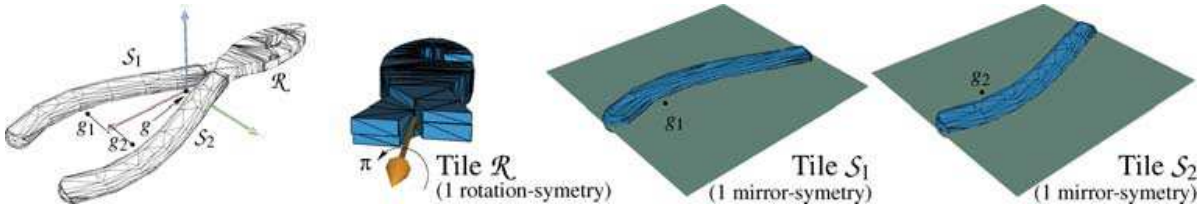


Fig. 9. Illustration of the constructive algorithm on a very simple example: from the symmetries of each of the 3 parts of the object, the symmetries of the whole object are recovered. Please note that no symmetry was ommitted in this Figure. In particular, tile $\mathcal{R}$ has only a rotational symmetry but no mirror symmetry. See text of Section 6.4 for a detailed explanation.

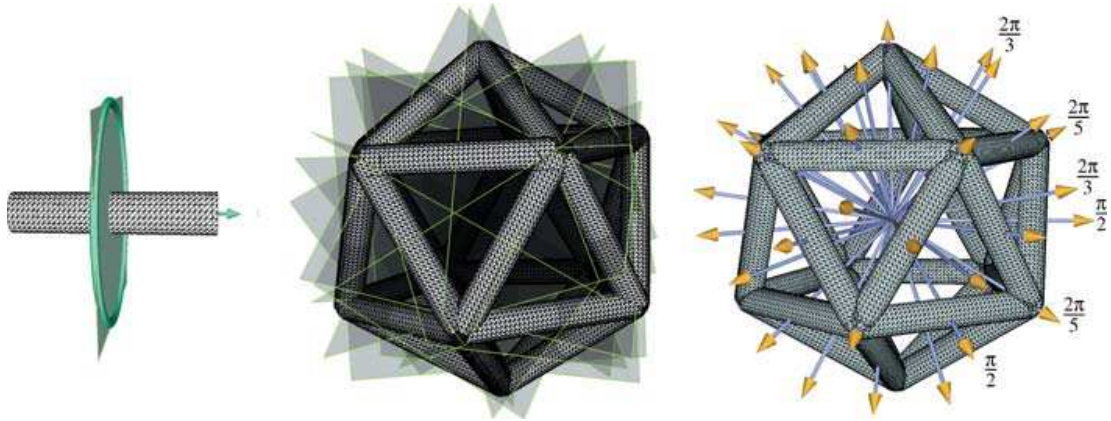


Fig. 10. A complex model which has the same group of symmetries as the icosahedron. The constructive algorithm successfully retrieves all 15 planes of mirror symmetries (*center*) and all 31 distinct axes of rotational symmetries (*right*) using the rotational and mirror symmetry of each tile (at *left*). The presence of 3−fold and 5−fold symmetries proves that our algorithm also detects symmetries which map a set of similar tiles onto itself through a complex permutation.

Branch 1, Step 2. The algorithm then tries to map $\mathcal{S}_2$ to itself. Once again, the only possible transform is the identity transform, and the recursion stops because all the tiles in the model have been used.
Branch 2, Step 1. The algorithm tries to match $\mathcal{S}_1$ to $\mathcal{S}_2$. The only compatible transform is the rotation around $\boldsymbol{X}$ of angle π.
Branch 2, Step 2. The algorithm then tries to match $\mathcal{S}_2$ to $\mathcal{S}_1$. Once again, the only compatible transform is the rotation around $\boldsymbol{X}$ of angle π, and the recursion stops because all the tiles in the model have been used.

Two symmetries have been found that map the shape onto itself, the identity transform and the rotation around $\boldsymbol{X}$ of angle π. Note that, although our algorithm can potentially create lots of branching, we prune branches that result in empty sets of transforms and, in practice, we only explore a small number of branches.

6.5 Application Scenarios

In order to illustrate the efficiency of the constructive algorithm, we show in this section various situations where this method is a valuable alternative to the single-shape algorithm.

6.5.1 *Application to an Agregation of Many Objects.* Figure 10 presents a complex model which has the same group of symmetries as an icosahedron. The constructive algorithm retrieves all the 31 distinct axis of rotational symmetries (Figure 10, *right*) as well as the 15 axis of plannar symmetries (Figure 10,

Table IV. Comparison of the costs of the single-shape algorithm presented in Section 5 to the cost of the constructive algorithm to find all 46 symmetries of the icosahedron shape displayed on Figure 10 at equivalent precision. Because the object is close to a sphere and because it has many symmetries, the constructive algorithm performs much better

Method	Single shape (order 10)	Constructive (order 4)
Moments calculation	500 sec	30×0.5 sec
Symmetry verification	46×55 sec	$30 \times 2 \times 1.5$ sec
Tile congruency	N/A	2 sec
Tile mappings	N/A	10 sec
Total	**50mn 30 sec**	**1mn 57 sec**

middle) of the shape, using the symmetries of each tile (Figure 10, *left*), which are 1 revolution symmetry and 1 mirror symmetry.

Conversely, directly applying the first algorithm on such a shape shows that $\mathcal{M}^2$ to $\mathcal{M}^8$ are extremely close to constant functions, making the extraction of directions an inaccurate process. The single-shape algorithm still correctly finds all the axis if using moments up to order 10, but this has some impact on computation times. Furthermore, the single-shape algorithm requires checking all of the symmetries found on the model which is a significant part of its computation time. This is not the case for the constructive algorithm because it relies on its knowledge of the symmetries of the tiles only. Because many symmetries exist for this model, the total computation time of the single-shape algorithm is therefore much higher. This is summarized in Table IV where we compare the computation times for both methods at equivalent precision (i.e., 10^{-4} radians).

6.5.2 *Finding Symmetries Inside Noncoherent Geometry.* There exist common situations where 3D scenes do not come as a set of closed separate objects but as an incoherent list of polygons. This happens, for instance, when retrieving geometric data from a Web site, mostly because a list of polygons constitutes a practical common denominator to all possible formats.

In such a case, applying the single-shape algorithm would certainly give the symmetries of the whole scene but if we are able to partition the set of polygons into adequate groups, that is, tiles to which we apply the constructive algorithm, we may be able to extract symmetric objects from the scene as well as the set of symmetries for the whole scene more rapidely as illustrated in Figure 10.

The gain in using the constructive algorithm to recover symmetries in the scene resides in the fact that, once tile symmetries have been computed, grouping them together and testing for symmetries in composed objects only adds a negligible cost which is not the case when we try to apply the single-shape algorithm to many possible groups of polygons or even to the entire scene itself.

The various issues in the decomposition of a raw list of polygons into intelligent tiles are beyond the scope of this article. In our case, tiles only need to be consistent with the symmetries. We propose the following heuristic to achieve this correctly for most scenes:

We define tiles as maximal sets of edge-connected polygons. To obtain them, we insert all vertices of the model into a KDTree and use this KDTree to efficiently recover which polygons share vertices up to a given precision and share an edge. By propagating connectivity information between neighboring polygons, we then build classes of edge-connected polygons, which we define to be our tiles. Figure 11 gives examples of such tiles for objects collected from the Web as a raw list of polygons.

Our simple heuristic approach of making tiles produced very good results on all scenes we tested and suffices for a proof of concept of the constructive algorithm. This is illustrated in Figure 11 where a lamp object and a chess game are shown along with their global symmetries. These symmetries were

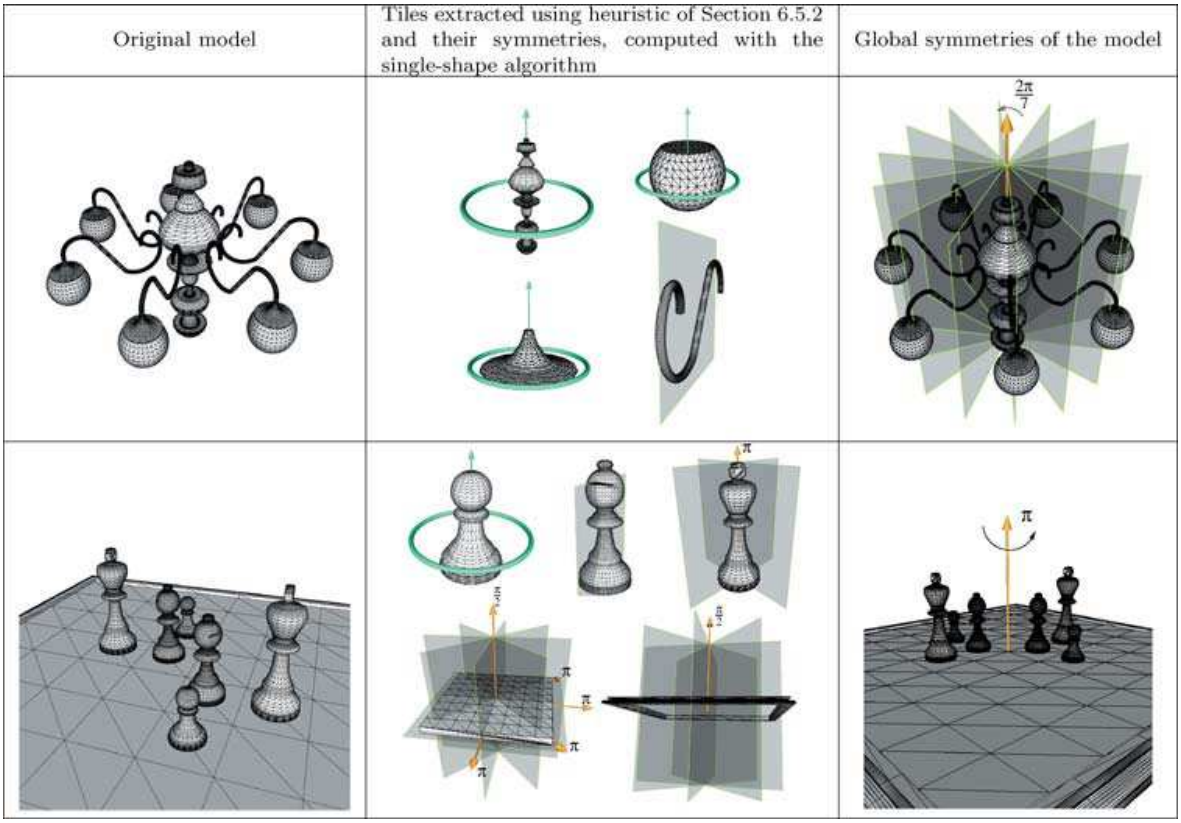


Fig. 11. Two models taken from the Web. From the raw list of polygons (left) our heuristic for scene partitionning extracts tiles before the single-shape algorithm computes the symmetries for each of them (center). Using this information, the constructive algorithm computes the symmetries of the whole model (right). *Top row*: A lamp object which has seven mirror symmetries and a 7−fold rotational symmetry. *Bottom row*: a chess board which is composed of pieces with very different symmetries but reveals to only have a single 2−fold symmetry around a vertical axis (Note: in this last model, once tiles have been identified, chess pieces were moved so as to obtain a model with at least one global symmetry).

computed from the symmetries of each of the subparts. These, in turn, were separately computed using the algorithm presented in Section 5.

Obviously, this application requires that constructed tiles be consistent with symmetries, that is, that it is possible to partition the scene into tiles which will map onto each other through the symmetries of the scene. This may not be easy with scanned models, for instance, nor in perturbated data. In such a case, our simple heuristic should be modified so as to base polygon neighborood relationships on proximity distances between polygons rather than vertex positions only. Doing so, cutting one tile into two parts and remeshing them independently, would have a high probability of producing the same original tile after reconstruction. If not, then the existance of a symmetry inside the model may become questionnable. Suppose, for instance, that the wrench in the step-by-step example (Section 6.4) gets split into tiles that are not exact symmetrical copies of one another, and that these two tiles are too far away to be merged into a single tile. Then the model is by nature not symmetric anymore which will also be the output of the constructive algorithm.

Table V. Computation Times (in seconds) for the Different Steps of our Algorithm, for the Models Shown in this Article

Model	Plier	Lamp	Chessboard
# polygons	1,940	39,550	24,942
# tiles	3	22	8
Computing moments*	0.9	18.2	15
Finding parameters	0.4	1.2	2.0
Checking candidates	2.3	7.4	7.9
Constructive algo.	0.001	0.05	0.01
Total	3.601	26.85	24.91

*Global computation time for moments of order 2 to 8.

6.6 Computation Cost

Computation times (in seconds) for the models shown in this article are given in Table V as well as the complexity of the models. They were measured on a machine equipped with a 2.4GHz processor with 512MB of memory. As expected, the cost of the computation of the moment functions and the cost of the verification of the candidates required by the first algorithm occupy the most important part of the total cost and depend on the model complexity. Conversely, finding the parameters of the symmetries (Section 5.2) as well as applying the constructive algorithm only depends on the number of these symmetries.

Regarding accuracy, both algorithms computed the axes of the symmetries with a maximum error of 10^{-4} radians, independently of shape complexity, in our tests.

7. APPLICATIONS

7.1 Geometry Compression and Instantiation

Our framework can be used for model compression at two different levels. (1) If a model exhibits symmetries, then it can be compressed by storing only the significant part of the model and using the symmetries to recreate the full model. (2) If a model contains multiple instances of the same part, then these parts can be instantiated. (see Figure 12).

Although complex models often do not present symmetries, symmetry-based compression can usually be used on some subparts of the model. The ability to express a model by explicitely storing the significant parts only while instancing the rest of the scene is provided by some recent 3D file formats such as X3D (see Table VI). We thus measure our compression ratios as the size of the X3D files before and after our two compression operations which we detail now.

The scene is first loaded as a raw collection of polygons, before being decomposed into tiles, using the heuristic presented in Section 6.5.2. We then compute symmetries and congruent descriptors for each tile. Computation times shown in Table VI present the average time needed to compute symmetries and congruent descriptors for a single tile. As the process of computing tile properties does not depend on the other tiles, it is an easily parallelizable process. The scene is then first compressed by instancing the tiles. Secondly, when storing each tile, we only store the minimum significant part of its geometry according to its symmetries. This part is extracted using the same algorithm we will present for remeshing a tile according to its symmetries in the next section. Note that compression rates shown on this table are computed using geometry informations only, that is, neither texturing nor material information are taken into account. Compression times shown in Table VI are the times needed to detect all classes of tile congruency.

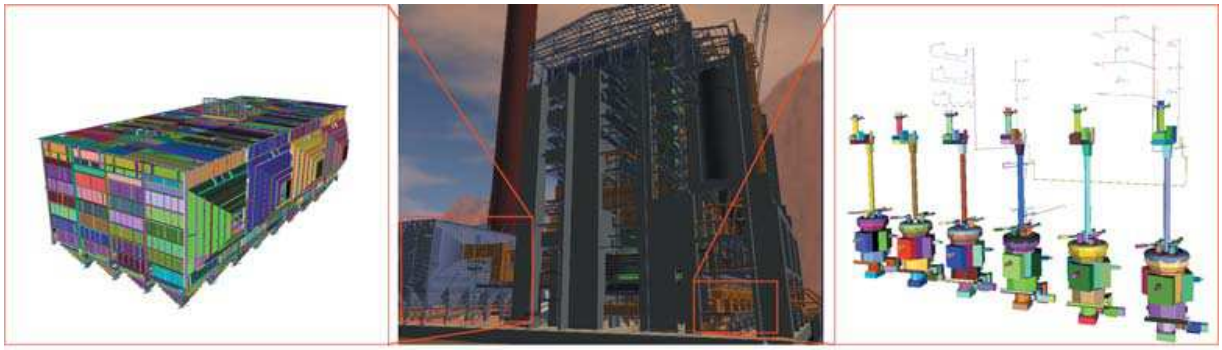


Fig. 12. Detecting symmetries and similarities between tiles created from a raw list of polygons allows us to compress geometric models in two ways: (1) by instancing similar tiles and (2) inside each symmetric tile, by instancing the part of the geometry which permits to reconstruct the whole tile. In such a big model as the powerplant (13 millions triangles), we achieve a compression ratio (ratio of geometry file size in X3D format) of 1:4.5. We show in this figure two subparts of the complete model. For each, we show the tiles computed by our heuristic (see Section 6.5) as well as the obtained compression ratio. The PowerPlant model is a courtesy of The Walkthru Project.

Table VI. Examples of Compression Rates Obtained Using our Symmetry Detection Method Coupled with the Congruency Descriptor. (See text in Section 7.1 for a detailed explanation.)

Model	Room	Plane	Studio	Powerplant
# polygons	39, 557	182, 224	515, 977	12, 748, 510
# tiles	851	480	5, 700	525, 154
av. computing tile properties (secs)	1.45	1.3	1.9	1.1
Compression time (secs)	7.2	9	14.6	311
Compression rate	1 : 2.7	1 : 8.3	1 : 3.5	1 : 4.5

7.2 Mesh Editing

It may be interesting, when an object contains symmetries, to remesh the object with respect to these symmetries. In order to do this, we proceed by first extracting the minimum part of the shape that can be reconstructed through each symmetry independently, then we apply the corresponding symmetry to each of them in order to get as many meshes of the shape which are consistent with each symmetry independently. The final step is to compute the union of all these meshes, merging identical vertices and adding new vertices at edge crossings. While not necessarily optimal, the obtained mesh is consistent with all symmetries of the shape.

Since a coherent remeshing allows for the establishment of a correspondence between model vertices, we have developed a proof-of-concept mesh editing system which allows the user to modify a 3D object under the constraints given by the symmetries of the original object. It appears that, under the constraint of too many symmetries, no vertices can be moved independently of the others, and the geometry is sometimes bound to scale about its center of gravity. Images collected from this program are displayed in Figure 13.

8. DISCUSSION

We discuss here a number of features of our technique as well as differences with existing approaches.

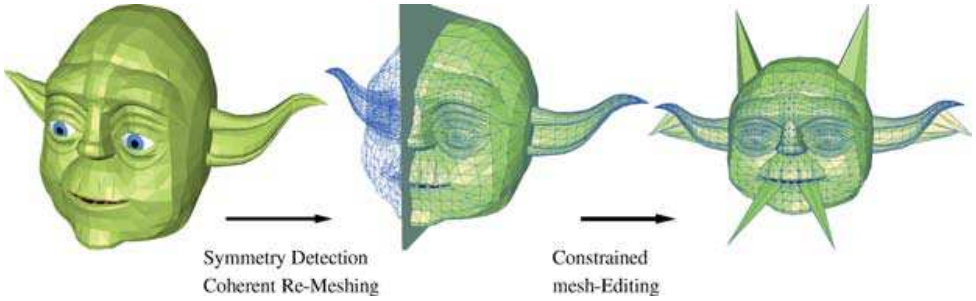


Fig. 13. Starting from an object in arbitrary orientation, we detect symmetries of the shape (in the figure, a planar symmetry) and use it to remesh the objects with respect to these symmetries. Then, a user can easily edit the mesh and modify it while keeping the symmetries of the initial shape.

Using Spherical Harmonics

Generalized moments are a central component of our system. As stated before, we do not compute these functions explicitly but we rather compute their coefficients in a spherical harmonics basis. As for the decomposition itself, any basis could be used. In particular, a well chosen basis of 3D monomials restricted to the unit sphere may also lead to a finite decomposition. Still, using spherical harmonics has many advantages, in particular, because we use the same coefficients computed once for different tasks throughout this article. (1) The expression of moment function as a sum of spherical harmonics provides an accurate detection of the potential axes of symmetries. This detection is made deterministic by finding the zero directions for the gradient of the moment functions. Such a computation is performed analytically from the 2nd order derivatives of the spherical harmonics, and thus does not introduce further approximation. (2) Computing symmetry parameters for the moment functions is made very easy by working on the spherical harmonic coefficients themselves. Since spherical harmonics are orthogonal and easily rotated, finding symmetries on the moment functions translates into simple relationships between the coefficients. (3) The spherical harmonic coefficients provide an effective shape congruency descriptor which we use to detect which tiles are identical up to an unknown isometric transform.

In summary, the use of spherical harmonics provides us a consistent framework throughout the whole process of our symmetry-finding algorithm.

Non Star-Shaped Objects

Whether the direct algorithm presented in Section 5 works for non star-shaped objects is a legitimate question. Our approach never relies on a spherical projection. Indeed, the moment functions, as expressed in Equations (1) and (5) are computed through an integration over the surface itself, possibly covering the same directions multiple times but with different values. Parts of a shape which correspond to a same direction during integration will not contribute the same into the various moment functions because of the varying exponent. By using various orders of moment functions in our symmetry detection process and in the computation of our shape congruency descriptor, we thus capture the geometry of non star-shaped objects as well. Some previous approaches [Kazhdan et al. 2004] achieved this by decomposing the shape into concentric spherical regions before doing a spherical integration which can be assimilated to convoluting the shape with 0-degree functions with concentric spherical support; Our technique is similar, but with another, kind of functions expressed into the form of the even moments. In summary, detecting symmetries on non star-shaped objects has no particular reason to fail which is illustrated by the result in Figure 2.

The second algorithm (for assembled objects) naturally works just as well for non star-shaped objects as illustrated by the examples in Figure 11.

Avoiding Dense Sampling

Previous methods that defined a continuous measure of symmetry([Zabrodsky et al. 1995; Kazhdan et al. 2004]) can theoretically compute both perfect and approximate symmetries. However, detecting symmetries using such methods involves a sampling step of the directions on the sphere, whose density must be adapted to the desired angular precision for the axis of the symmetry.

The work of Kazhdan et al. [2004] leads to impressive results concerning the improvment on the shape matching process. However, relying on this technique to obtain accurate symmetries with high angular precision requires a time-consuming step for the construction of the symmetry descriptors. According to the presented results, the time needed to compute reflective, 2-fold, 3-fold, 4-fold, 5-fold, and axial symmetry information for a spherical function of bandwidth $b = 16$ is 0.59 seconds. As stated in the article [Kazhdan et al. 2004], the number of samples taken on the sphere is $O(b^2)$ (i.e., approximately 10^3 sample directions) which is insufficent to reach a high angular precision equivalent to the one obtained with our method: reaching a precision of 10^{-4} radians would require approximately 10^9 sample directions. This would theoretically increase the computation time to approximately $0.59 \times 10^9/10^3 = 5.9\ 10^5$ seconds, making the method inefficient for this task.

In contrast, our method does not rely on a dense sampling of directions to find symmetries but on the computation of a fixed number of surface integrals which—thanks to the Gauss integration used—provides an extremely accurate approximation of the spherical harmonic coefficients of the moment functions. From there on, no further approximation is introduced in the computation of the directions of the candidate symmetries which lets us achieve an excellent angular precision at a much lower cost.

Furthermore, the cost of our algorithm does not rely on assumptions about the expected results. The method of Kazhdan et al. [2004] indeed computes symmetry descriptors for each kind of searched symmetry. Our method in turn computes all directions of possible symmetries and then checks back on the shape of the obtained candidates.

9. CONCLUSIONS

We have presented an algorithm to automatically retrieve symmetries for geometric shapes and models. Our algorithm efficiently and accurately retrieves all symmetries from a given model, independently of its tesselation.

We use a new tool, the generalized moment functions, to identify candidates for symmetries. The validity of each candidate is checked against the original shape using a geometric measure. Generalized moments are not computed directly: instead, we compute their spherical harmonic coefficients using an integral expression. Having an analytical expression for the generalized moment functions and their gradients, our algorithm finds potential symmetry axes quickly and with good accuracy.

For composite shapes assembled from simpler elements, we have presented an extension of this algorithm that works by first identifying the symmetries of each element, then sets of congruent elements. We then use this information to iteratively build the symmetries of the composite shape. This extension is able to handle complex shapes with better accuracy since it pushes the accuracy issues down to the scale of the tiles.

Future Work

The constructive algorithm presented in Section 6 automatically detects instantiation relationships between tiles into a composite shape.

We are currently developing a constructive instantiation algorithm which iteratively collates similar tiles into instances, checking at each step that the relative orientation of each tile with respect to each already constructed instance is preserved.

This algorithm requires the symmetries of the tiles, and maintaining the symmetries of the instances found so far. For this, we use our shape congruency metric, our algorithm for finding symmetries of single shapes, and our algorithm for finding symmetries on composite shapes.

APPENDIX (PROOFS)

PROOF OF THEOREM 1. Let A be an isometric transform which lets a shape $\mathcal{S}$ be globally unchanged. We have:

$$\begin{aligned}
\forall \boldsymbol{\omega} \quad \mathcal{M}^{2p}(A\boldsymbol{\omega}) &= \int_{\boldsymbol{s}\in\mathcal{S}} \|\boldsymbol{s}\times A\boldsymbol{\omega}\|^{2p}\,\mathrm{d}\boldsymbol{s} \\
&= \int_{\boldsymbol{t}\in A^{-1}\mathcal{S}} \|A\boldsymbol{t}\times A\boldsymbol{\omega}\|^{2p}|\det A|\,\mathrm{d}\boldsymbol{t} \\
&= \int_{\boldsymbol{t}\in A^{-1}\mathcal{S}} \|\boldsymbol{t}\times \boldsymbol{\omega}\|^{2p}\,\mathrm{d}\boldsymbol{t} \\
&= \mathcal{M}^{2p}(\boldsymbol{\omega})
\end{aligned}$$

At line 2, we change variables and integrate over the surface transformed by A^{-1}. At line 3, an isometric transform is a unit transform and so, its determinant is ± 1 and thus vanishes. The cross product is also left unchanged by applying an isometric transform to each of its terms. Line 4: because $A\mathcal{S}=\mathcal{S}$, we also have $\mathcal{S}=A^{-1}S$. The isometric transform A is thus also a symmetry of the $\mathcal{M}^{2p}$ moment functions.

Let A be an isometric transform with axis $\boldsymbol{v}$, and suppose that A is a symmetry of $\mathcal{M}^{2p}$. Let $\boldsymbol{d_v}$ be the direction of steepest descent of function $\mathcal{M}^{2p}$ around direction $\boldsymbol{v}$. Because A is a symmetry of $\mathcal{M}^{2p}$, we have:

$$\boldsymbol{d}_{A\boldsymbol{v}} = A\boldsymbol{d_v} = \boldsymbol{d_v}. \tag{13}$$

If A is a rotation, this is impossible because $\boldsymbol{d_v}\perp\boldsymbol{v}$. Moreover, for all directions ω, we have $\mathcal{M}^{2p}(-\omega)=\mathcal{M}^{2p}(\omega)$ and thus:

$$\boldsymbol{d}_{-\boldsymbol{v}} = -\boldsymbol{d_v}. \tag{14}$$

So, if A is a symmetry, we have $A\boldsymbol{v}=-\boldsymbol{v}$. From Equations (13) and (14), we get $\boldsymbol{d_v}=-\boldsymbol{d_v}$ which is impossible.

In both cases, $\mathcal{M}^{2p}$ can not have a direction of steepest descent in direction $\boldsymbol{v}$. Because $\mathcal{M}^{2p}$ is infinitely derivable, this implies that $\boldsymbol{\nabla}\mathcal{M}^{2p}(\boldsymbol{v})=0$ □

PROOF OF PROPERTY 4. Let $\mathcal{S}$ and $\mathcal{R}$ be two shapes, identical up to an isometric transform. Let J be an isometric transform such that $J\mathcal{S}=\mathcal{R}$. Let T be the translation of vector $-u_{\mathcal{S}}$ with $\boldsymbol{u}_{\mathcal{S}}=\boldsymbol{g}_{\mathcal{S}}-\boldsymbol{g}$ with $\boldsymbol{g}_{\mathcal{S}}$ as the center of mass of $\mathcal{S}$, and $\boldsymbol{g}$ the origin of the coordinate system into which J is applied.

— Let $A\in G_S$ be a symmetry of $\mathcal{S}$ such that $A\boldsymbol{u}_{\mathcal{S}}=\boldsymbol{u}_{\mathcal{S}}$. We have $AT\mathcal{S}=T\mathcal{S}$ (the symmetry A operates in the coordinate system centered on $\boldsymbol{g}_{\mathcal{S}}$). Let $K=JT^{-1}AT$. Then

$$\begin{aligned}
K\mathcal{S} &= JT^{-1}AT\mathcal{S} \\
&= JT^{-1}T\mathcal{S} \\
&= J\mathcal{S} \\
&= \mathcal{R}
\end{aligned}
\qquad\text{and}\qquad
\begin{aligned}
K\mathbf{0} &= JT^{-1}AT\mathbf{0} \\
&= JT^{-1}A(-\boldsymbol{u}_{\mathcal{S}}) \\
&= JT^{-1}(-\boldsymbol{u}_{\mathcal{S}}) \\
&= J\mathbf{0}=0
\end{aligned}$$

By construction K is a rigid transform and conserves distances. It maps the origin onto itself. K is thus an isometric transform. Furthermore, K maps $\mathcal{S}$ to $\mathcal{R}$.

— Let K be an isometric transform such that $K\mathcal{S} = \mathcal{R}$. Let us choose $A = TJ^{-1}KT^{-1}$. This choice leads to $K = JT^{-1}AT$. Moreover:

$$\begin{aligned} AT\mathcal{S} &= TJ^{-1}KT^{-1}T\mathcal{S} \\ &= TJ^{-1}KS \\ &= T\mathcal{S} \end{aligned} \qquad \text{and} \qquad \begin{aligned} A\boldsymbol{u}_{\mathcal{S}} &= TJ^{-1}KT^{-1}\boldsymbol{u}_{\mathcal{S}} \\ &= TJ^{-1}K2\boldsymbol{u}_{\mathcal{S}} \\ &= T2\boldsymbol{u}_{\mathcal{S}} = \boldsymbol{u}_{\mathcal{S}} \end{aligned}$$

and

$$\begin{aligned} A\mathbf{0} &= TJ^{-1}KT^{-1}\mathbf{0} \\ &= TJ^{-1}K\boldsymbol{u}_{\mathcal{S}} \\ &= TJ^{-1}(g_{\mathcal{R}} - \boldsymbol{g}) \\ &= T(-\boldsymbol{u}_{\mathcal{S}}) \\ &= \mathbf{0} \end{aligned}$$

By construction A is affine and conserves distances. It maps $\mathbf{0}$ onto $\mathbf{0}$. A is thus an isometric transform. A is also a symmetry of $\mathcal{S}$ which verifies $A\boldsymbol{u}_{\mathcal{S}} = \boldsymbol{u}_{\mathcal{S}}$.

— The set of isometries which map $\mathcal{S}$ to $\mathcal{R}$ is therefore the set of functions K of the form $K = JT^{-1}AT$, where $A \in G_{\mathcal{S}}$ is a symmetry of $\mathcal{S}$ such that $A(\boldsymbol{g} - \boldsymbol{g}_{\mathcal{S}}) = (\boldsymbol{g} - \boldsymbol{g}_{\mathcal{S}})$.

□

PROOF OF EQUATION 3. We compute the decomposition of function $\theta \longmapsto \sin^{2p}\theta$ into zonal spherical harmonics. We prove that this decomposition is finite, and give the values of the coeficients.

By definition [Hobson 1931], we have:

$$\begin{aligned} Y_L^0(\theta,\varphi) &= \sqrt{\frac{2L+1}{4\pi}} P_L(\cos\theta) \\ &= \sqrt{\frac{2L+1}{4\pi}} \frac{(-1)^L}{2^L L!} \frac{d^L}{dx^L}\left[(1-x^2)^L\right](\cos\theta) \end{aligned}$$

where P_k is the Legendre polynomial of order k. Because the set of Legendre polynomials $P_0, P_1, ..., P_n$ is a basis for polynomials of order not greater than n, function $\theta \longmapsto \sin^{2p}\theta = (1-\cos^2\theta)^p$ can be uniquely expressed in terms of $P_L(\cos\theta)$. The decomposition of $\theta \longmapsto \sin^{2p}\theta$ is thus finite and has terms up to Y_{2p}^0 at most.

Let's compute them explicitely:

$$\begin{aligned} \frac{d^L}{dx^L}\left[(1-x^2)^L\right] &= \frac{d^L}{dx^L}\sum_{k=0}^{L}(-1)^{L-k}x^{2L-2k}C_L^k \\ &= (-1)^L\frac{d^L}{dx^L}\sum_{k=0}^{L}(-1)^k x^{2k}C_L^k \\ &= \sum_{L\le 2k\le 2L}(-1)^{L+k}C_L^k 2k(2k-1)...(2k-L+1)x^{2k-L} \\ &= \sum_{L\le 2k\le 2L}(-1)^{L+k}C_L^k\frac{(2k)!}{(2k-L)!}x^{2k-L} \end{aligned}$$

So:

$$Y_L^0(\theta,\varphi) = \sqrt{\frac{2L+1}{4\pi}} \sum_{L\le 2k\le 2L} \frac{(-1)^k}{2^L L!} C_L^k \frac{(2k)!}{(2k-L)!} \cos^{2k-L}\theta$$

The coeficients of the decomposition we are interested in are thus:

$$\int_{\theta=0}^{\pi}\int_{\varphi=0}^{2\pi} Y_L^0(\theta,\varphi)\sin^{2p}\theta\sin\theta d\theta d\varphi = 2\pi\sqrt{\frac{2L+1}{4\pi}} \sum_{L\le 2k\le 2L} \frac{(-1)^k}{2^L L!} C_L^k \frac{(2k)!}{(2k-L)!} I_{2k-L}^p \tag{15}$$

where integrals I_m^p are defined by:

$$I_m^p = \int_{\theta=0}^{\pi} \sin^{2p+1}\theta\cos^m\theta d\theta$$

First, $I_m^p = 0$ for all odd m because the integrand in antisymetric around $x = \pi/2$. Then, if m is even:

$$\begin{aligned} I_m^p &= \underbrace{\left[\frac{1}{2p+2}\sin^{2p+2}\theta\cos^{m-1}\theta\right]_0^{\pi}}_{0} + \frac{m-1}{2p+2}\int_0^{\pi}\sin^{2p+3}\theta\cos^{m-2}\theta d\theta \\ &= \frac{m-1}{2p+2} I_{m-2}^{2p+3} \\ &= \frac{(m-1)(m-3)\ldots 1}{(2p+2)(2p+4)\ldots(2p+m)}\int_o^{\pi}\sin^{2p+m+1}\theta d\theta \end{aligned}$$

Let J_q be the integral defined by

$$J_q = \int_0^{\pi}\sin^{2q+1}\theta d\theta.$$

We have

$$\begin{aligned} J_q &= \underbrace{[-\cos\theta\sin^{2q}\theta]_0^{\pi}}_{0} + 2q\int_0^{\pi}\cos^2\theta\sin^{2q-1}\theta d\theta \\ &= 2qJ_{q-1} - 2qJ_q \end{aligned}$$

Therefore

$$\begin{aligned} J_q &= \frac{2q}{2q+1}J_{q-1} \\ &= \frac{2q(2q-2)\ldots 2}{(2q+1)(2q-1)\ldots 3}J_0 \\ &= \frac{2^{2q+1}(q!)^2}{(2q+1)!} \end{aligned}$$

For m even, we can take $m = 2r$ and $q = p + r$; we get:

$$\begin{aligned} I_{2r}^p &= \frac{(2r)!p!}{2^r r! 2^r (p+r)!}\frac{2^{2p+2r+1}(p+r)!^2}{(2p+2r+1)!} \\ &= \frac{(2r)!p!2^{2p+1}(p+r)!}{r!(2p+2r+1)!} \end{aligned} \tag{16}$$

From Equation (15), we deduce that, for L odd,

$$\int\int Y_L^0(\theta,\varphi)\sin^{2p}\theta\sin\theta d\theta d\varphi = 0.$$

For L even, we set $L = 2l$. Using $r = k - l$ to match Equation (16) in Equation (15), we get:

$$\begin{aligned}
S_p^l &= \int\int Y_{2l}^0(\theta,\varphi)\sin^{2p}\theta\sin\theta d\theta d\varphi \\
&= 2\pi\sqrt{\frac{4l+1}{4\pi}}\sum_{2l\le 2k\le 4l}\frac{(-1)^k}{2^{2l}(2l)!}C_{2l}^k\frac{(2k)!}{(2k-2l)!}\frac{(2k-2l)!p!2^{2p+1}(p+k-l)!}{(k-l)!(2p+2k-2l+1)!} \\
&= \frac{\sqrt{(4l+1)\pi}}{2^{2l}(2l)!}\sum_{l\le k\le 2l}(-1)^k C_{2l}^k\frac{(2k)!p!2^{2p+1}(p+k-l)!}{(k-l)!(2p+2k-2l+1)!} \\
&= \frac{\sqrt{(4l+1)\pi}}{2^{2l}}\sum_{l\le k\le 2l}(-1)^k\frac{(2k)!p!2^{2p+1}(p+k-l)!}{k!(2l-k)!(k-l)!(2p+2k-2l+1)!}
\end{aligned}$$

□

REFERENCES

ATTALAH, M. J. 1985. On symmetry detection. *IEEE Trans. Comput. 34*, 663–666.

BRASS, P. AND KNAUER, C. 2004. Testing congruence and symmetry for general 3-dimensional objects. *Comput. Geom. Theory Appl. 27*, 1, 3–11.

HIGHNAM, P. T. 1985. Optimal algorithms for finding the symmetries of a planar point set. Tech. Rep. CMU-RI-TR-85-13 (Aug). Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

HOBSON, E. W. 1931. *The Theory of Spherical and Ellipsoidal Harmonics*. Cambridge University Press, Cambridge, UK.

IVANIC, J. AND RUEDENBERG, K. 1996. Rotation matrices for real spherical harmonics, direct determination by recursion. *J. Phys. Chem. A. 100*, 6342–6347. (See also *Additions and corrections* in vol. 102, No. 45, 9099-9100).

JIANG, X.-Y. AND BUNKE, H. 1991. Determination of the symmetries of polyhedra and an application to object recognition. In *Proceedings of the International Workshop on Computational Geometry—Methods, Algorithms and Applications (CG '91)*. Lecture Notes in Computer Science, vol. 553. Springer, London, UK, 113–121.

KAZHDAN, M. M., FUNKHOUSER, T. A., AND RUSINKIEWICZ, S. 2003. Rotation invariant spherical harmonic representation of 3D shape descriptors. In *Proceedings of the 2003 Eurographics/ACM Siggraph Symposium on Geometry Processing (SGP '03)*. Eurographics Association, Aire-la-Ville, Switzerland, 167–175.

KAZHDAN, M. M., FUNKHOUSER, T. A., AND RUSINKIEWICZ, S. 2004. Symmetry descriptors and 3D shape matching. In *Proceedings of the 2004 Eurographics/ACM Siggraph Symposium on Geometry Processing (SGP '04)*. Eurographics Association, Aire-la-Ville, Switzerland.

KNUTH, D. E., MORRIS, JR., J. H., AND PRATT, V. R. 1977. Fast pattern matching in strings. *SIAM J. Comput. 6*, 2, 323–350.

MINOVIC, P., ISHIKAWA, S., AND KATO, K. 1993. Symmetry identification of a 3-D object represented by octree. *IEEE Trans. Patt. Analy. Mach. Intell. 15*, 5, 507–514.

PRINCE, E. 2004. *Mathematical Techniques in Crystallography and Materials Science*, 3rd Ed. Springer, Berlin, Germany.

RAMAMOORTHI, R. AND HANRAHAN, P. 2004. A signal-processing framework for reflection. *ACM Trans. Graph. 23*, 4, 1004–1042.

SUN, C. AND SHERRAH, J. 1997. 3D symmetry detection using extended Gaussian image. *IEEE Trans. Patt. Analy. Mach. Intell. 19*, 2 (Feb.), 164–168.

WOLTER, J. D., WOO, T. C., AND VOLZ, R. A. 1985. Optimal algorithms for symmetry detection in two and three dimensions. *Visual Comput. 1*, 37–48.

ZABRODSKY, H., PELEG, S., AND AVNIR, D. 1995. Symmetry as a continuous feature. *IEEE Trans. Patt. Analy. Mach. Intell. 17*, 12, 1154–1166.

Received August 2005; revised December 2005; accepted January 2006

Edge-preserving Multiscale Image Decomposition based on Local Extrema

Kartic Subr* INRIA / Grenoble University
Cyril Soler† INRIA / Grenoble University
Frédo Durand ‡ MIT CSAIL

(a) Input (b) Fine features boosted (c) Coarse features boosted (d) Scanline plots

Figure 1: *Our multiscale decomposition of image (a) allows detail to be extracted based on spatial scale rather than contrast and preserves edges. (b) Boosting fine scale features increases the contrast of the pattern on the vase. (c) Boosting coarse scale contrast and suppressing fine features reduces the contrast of the pattern, while increasing the contrast of the vase with its background. (d) Scanline plots (rows indicated using arrows in (a), (b) and (c)), illustrating the effect of the two equalizations (b) and (c). The dashed lines in the plots show two examples of edges that have been preserved.*

Abstract

We propose a new model for detail that inherently captures *oscillations*, a key property that distinguishes textures from individual edges. Inspired by techniques in empirical data analysis and morphological image analysis, we use the local extrema of the input image to extract information about oscillations: We define detail as oscillations between local minima and maxima. Building on the key observation that the spatial scale of oscillations are characterized by the density of local extrema, we develop an algorithm for decomposing images into multiple scales of superposed oscillations.

Current edge-preserving image decompositions assume image detail to be low contrast variation. Consequently they apply filters that extract features with increasing contrast as successive layers of detail. As a result, they are unable to distinguish between high-contrast, fine-scale features and edges of similar contrast that are to be preserved.We compare our results with existing edge-preserving image decomposition algorithms and demonstrate exciting applications that are made possible by our new notion of detail.

Keywords: image decomposition, computational photography

1 Introduction

A variety of applications in computational photography require a decomposition of an image into different scales. Traditional approaches that use linear bases have evolved to accommodate the need for respecting strong edges. Recent definitions of scales are usually based on spatial scale definitions combined with a notion on the range to differentiate strong edges [Tomasi and Manduchi 1998; Durand and Dorsey 2002; Farbman et al. 2008; Lischinski et al. 2006; Choudhury and Tumblin 2005]. Current approaches share a common notion of an edge– large gradients, or large value differences, where the definition of large might depend on the application. However, this notion of an edge makes it challenging to capture fine details or textures that have fine spatial scale but high contrast. For example, in Figure 1(d), some edges to be preserved are lower contrast than oscillations to be smoothed. Extracting the white dots on the vase as detail requires aggressive smoothing of gradients, which would also blur single edges that are to be preserved (see Fig. 2). This distinction between edges and oscillations raises challenges in defining fully multiscale decompositions because the interplay between spatial and edge consideration leads to unexpected results, as shown by Farbman et al. [2008]

We propose a novel non-linear image decomposition that effectively extracts fine-scale features, regardless of their contrast, as detail and yet preserves softer salient edges in the base layer. In contrast to previous approaches that rely on magnitudes of pixel differences at their heart, our approach captures local image oscillations by considering local image extrema. A fine-scale texture is characterized by rapid oscillations (see Fig. 1) between minima and maxima. Furthermore, the oscillation between extrema provide critical information that permit the distinction of individual edges from oscillations. We obtain a multiscale decomposition by recursively smoothing the image while also progressively coarsening the scale at which extrema are detected.

1.1 Related work

Several image decomposition techniques have been proposed. Strategies that use linear filters [Burt and Adelson 1983; Rahman and Woodell 1997; Pattanaik et al. 1998] produce halo artifacts at edges and have been succeeded by non-linear filters that preserve strong edges– a popular choice being the bilateral filter [Tomasi and Manduchi 1998; Durand and Dorsey 2002; Choudhury and Tumblin 2005]. Bae et al. [2006] used the bilateral filter to separate images into low- and high-contrast features and manipulated the layers independently to enhance photographic look. Fattal et

*e-mail: Kartic.Subr@inrialpes.fr
†e-mail: Cyril.Soler@inrialpes.fr
‡e-mail: fredo@mit.edu

al. [2007] presented a technique to enhance shape and surface details of objects using bilaterally filtered representations of a set of differently lit images. Our goal is to extract from a single image, at each scale, the finest spatial oscillations as detail without assuming them to be low-contrast oscillations.

Two approaches have been proposed for multiscale decompositions using the bilateral filter. One strategy is to progressively increase the width of the range and spatial Gaussian through the coarsening process. Chen et al. [2007] used this technique to construct a *bilateral pyramid* for progressive video abstraction. Another strategy [Fattal et al. 2007] recursively applies the bilateral filter to the smoothed versions of the input image. This strategy decreases the width of the range-Gaussian during successive iterations so that edges from preceding smoothing operations are not blurred during the coarsening.

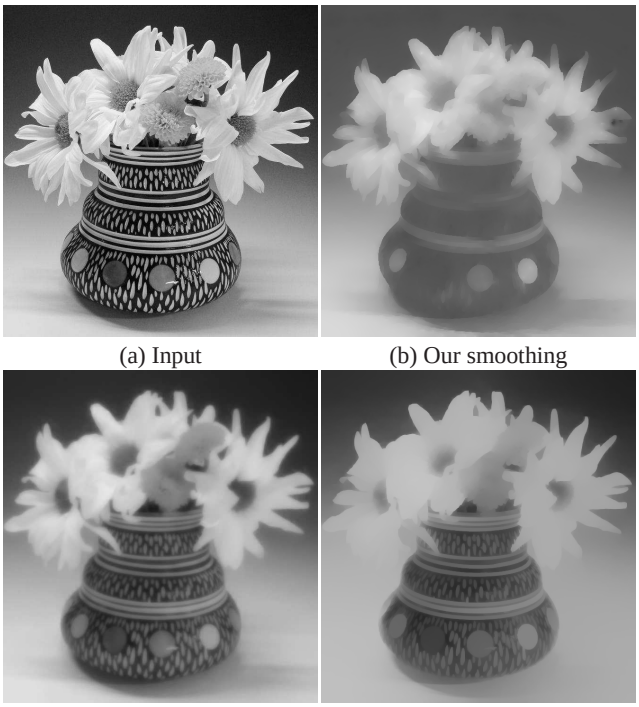
In recent work, Farbman et al. [2008] pointed out that, while the bilateral filter is effective at smoothing out low amplitude noise at a fine scale, multiscale decompositions using the bilateral filter suffer from a variety of problems. Progressive widening of the range and spatial Gaussians through the coarsening process was shown to produce halo artifacts at strong edges. To overcome some problems of using the bilateral filter in a multiscale decomposition, Farbman et al. [2008] proposed a filter that smoothes an input image I by computing an image that is as close to I as possible while being smooth everywhere except at regions where the gradient of I is large. They used a weighted least squares filter, originally used to control ringing during deblurring of noisy images [Lagendijk et al. 1988]. The nature of this optimization makes it impossible to preserve salient edges with lower contrast than the texture that is to be smoothed.

In summary, smoothing filters currently used in image decomposition algorithms assume detail is low-contrast. As a result, local variation at different contrast levels are extracted as successive layers of detail. Such layers of detail do not necessarily represent fine-scale spatial variation.

A notable exception, for 1D data, is empirical mode decomposition [Huang 1998]— a powerful data analysis tool originally proposed to decompose nonlinear, nonstationary signals into their intrinsic modes of oscillations. The decomposition is achieved by iterative removal of the finest intrinsic oscillations as indicated by local extrema. This technique is popularly used on 1D data that do not contain sharp discontinuities. A few attempts at extending the technique to image decomposition [Nunes et al. 2003; Liu and Peng 2005; Damerval et al. 2005] have uncovered a number of difficulties. One formidable challenge that has not been addressed is the need to respect sharp edges. Another drawback of empirical mode decomposition is its poor handling of signals where oscillations at different scales occur as bursts, in parts of the domain (the problem of intermittency [Li et al. 2005]).

1.2 Contributions

We introduce novel definitions, based on local extrema, for edges and detail that permit the distinction between highly contrasted texture and single edges. Using these definitions we develop an edge-preserving smoothing algorithm that allows fine scale detail to be extracted regardless of contrast. We perform an edge-preserving multiscale decomposition by recursively applying the smoothing algorithm on the base layer. The decomposition corresponds to features at different spatial scales with salient edges being preserved. We compare our approach with existing decompositions and demonstrate its effectiveness using applications. Figure 4 places our novel algorithm in the context of existing approaches.



(a) Input (b) Our smoothing

(c) WLS Filter ($\lambda = 13$, $\alpha = 0.2$) (d) WLS Filter ($\lambda = 13$, $\alpha = 1.2$)

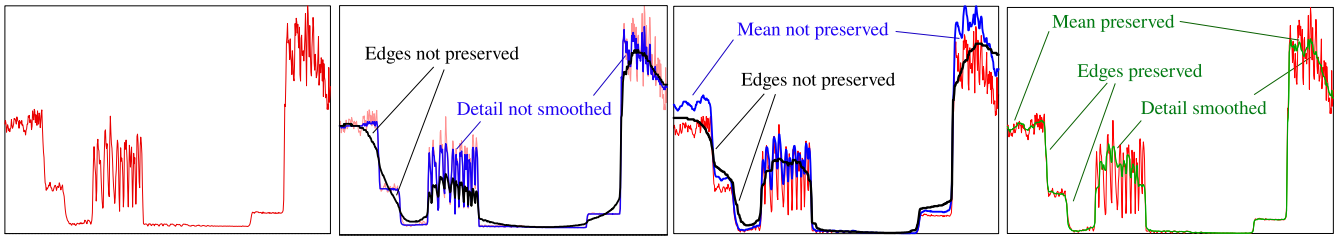
Figure 3: *The ubiquitous notion of edges as pixels with large gradients does not allow disambiguation between fine-scale features and edges that are to be preserved, as shown by this example.* (a) *The contrast of the pattern on the flower vase is greater than across the edges of the soft shadows and petal boundaries.* (b) *Using our smoothing algorithm, the pattern is extracted as detail because of its fine scale, while coarser soft shadow- and petal-boundaries are preserved.* (c) *The weighted least square (WLS) filter does not smooth the pattern if fidelity to strong gradients is retained.* (d) *On the other hand, the WLS filter necessarily blurs softer edges even though they are coarse-scale features while smoothing the pattern on the vase.*

2 Extrema-based multiscale decomposition

We present a novel smoothing algorithm which effectively smoothes highly contrasted oscillations while preserving salient edges. By applying this algorithm recursively on the smoothed image, we compute a *multiscale decomposition* of an input image into layers at different scales of coarseness. In comparison with existing edge-preserving multiscale decompositions, our algorithm significantly increases the ability to distinguish high-contrast texture from a dense field of edges.

Our notion of detail inherently captures repetitive variation of intensity, which we term *oscillations*. Locally, the amplitudes of oscillations represent contrast while their spatial-frequencies represents fineness in scale. Our goal is to smooth fine-scale oscillations, or *detail*, regardless of their amplitudes (see Fig 6). We extract the locally finest-scale oscillations as detail using a single smoothing operation, and obtain a multiscale decomposition by progressive smoothing. During successive smoothing operations on the residual, we coarsen the scale at which extrema are detected.

Inspired by empirical mode decomposition and morphological image filters, we examine the *local extrema* of the input image to detect oscillations. Empirical decomposition does not preserve edges



(a) Input (b) Bilateral Filter (c) WLS filter (d) Our method

Figure 2: *Intensity plots along a scanline of an input image are shown with three filtered versions: (b) Bilateral filtering with a conservative (blue) and aggressive (black) range parameter values ; (c) Gradient-based edge preserving smoothing technique (WLS filtering [2008]) with larger (blue) and smaller (black) gradient preserving parameter values; (d) Our smoothing filter. While existing techniques (b) and (c) are effective in smoothing variation with small amplitude (blue), they necessarily blur edges (black) that have smaller magnitudes of gradients than the oscillations to be smoothed. Our smoothing algorithm smoothes large oscillations and strictly preserves edges (green), without the need for careful selection of parameter values.*

	Definition of Detail	*Definition of Edge*	*Assumption*
Bilateral Filter	Low contrast variation	Large intensity difference	Texture is low-contrast
WLS Filter	Low contrast variation	Large gradient	Magnitude of gradient is larger at edges
BEMD	Fine-scale spatial oscillations	No notion of edge	Means of oscillations are smooth
Our Algorithm	Fine-scale spatial oscillations	High variance in range values of neighboring local extrema	Detail is oscillations between local extrema

Figure 4: *Comparison of our approach with three existing techniques for image decomposition: Bilateral filtering [Fattal et al. 2007] , weighted least squares (WLS) filtering [Farbman et al. 2008] and bidimensional empirical mode decomposition (BEMD) [Huang 1998] .*

while morphological operations do not preserve shape [Serra and Vincent 1992]. We exploit information provided by local extrema about the oscillations in the image and preserve both— edges and shape. Our algorithm is based on two key observations: (1) Detail (even if high-contrast) is characterized by a large density of local extrema; (2) salient edges (even if low-contrast) are characterized by a large variation in their neighboring extremal values.

Using local extrema, rather than contrast, to characterize detail provides two important benefits. First, we make no *a priori* assumptions on the dynamic range of the input image or on the amplitude of the oscillations. Second, we obtain the local scale of oscillations independent of contrast. Progressive coarsening of the scale at which extrema are detected results in layers with oscillations at different scales. Also, by recursively removing detail, the degrees of coarseness in the multiscale decomposition are likely to capture the inherent superimposed scales of oscillation in the input image.

For simplicity, we describe our algorithm for an input grayscale image I. Similar to existing decomposition techniques, we perform the decomposition on the luminance channel for color images. We denote image-space coordinates (x, y) with boldface letters. Thus $I(\mathbf{p})$ is the intensity of the given grayscale image I at pixel $\mathbf{p}$.

2.1 Smoothing

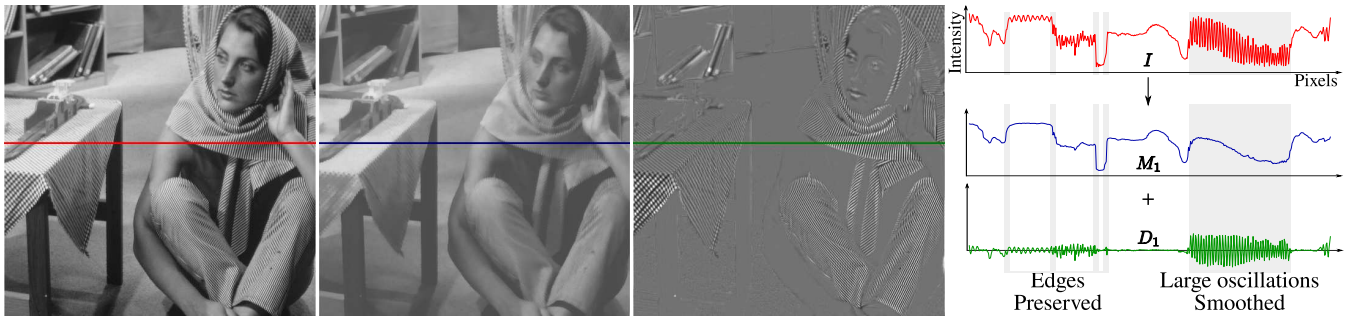
We define *detail* as *oscillations between local minima and maxima* (see Fig 5). We extract detail by subtracting a smoothed image, that we call the *mean*, from the input. The smoothing algorithm uses the local extrema to detect oscillations at their finest scale, locally. By interpolating the minima and maxima independently, we construct two *extremal envelopes*, that sandwich the data, and propagate information about local oscillations to all pixels in the image. The average of the two interpolants, evaluated at each pixel, provides an estimate of the local mean about which the oscillations occur. To ensure that the mean respects edges in the input image, the interpolants need to be edge preserving in the traditional sense that they retain fidelity to the input at strong gradients.

Our smoothing algorithm consists of three steps: (1) Identification of local minima and local maxima of I; (2) Interpolation of the local minima and maxima to compute minimal and maximal extremal envelopes respectively; (3) computation of the smoothed mean M as the average of the extremal envelopes. Figure 5 illustrates the three steps of our smoothing algorithm by plotting 1D slices of the Barbara input image (red), its extrema, extremal envelopes (blue and magenta) and smoothed mean (black). The detail layer is obtained as $D = I - M$.

Extrema location: We use a simple test for locating image maxima. Pixel $\mathbf{p}$ is reported as a maxima (resp. minima) if *at most* $k-1$ elements in in the $k \times k$ neighborhood around $\mathbf{p}$ are greater (resp. smaller) than the value at pixel $\mathbf{p}$. Oscillations whose maxima are detected by using a $k \times k$ kernel have wavelengths of at least $k/2$ pixels. Intuitively, using a large kernel overlooks the detection of fine oscillations. We start with $k = 3$ and increase the kernel size for multiscale smoothing, after extracting fine oscillations (see Sec. 2.2).

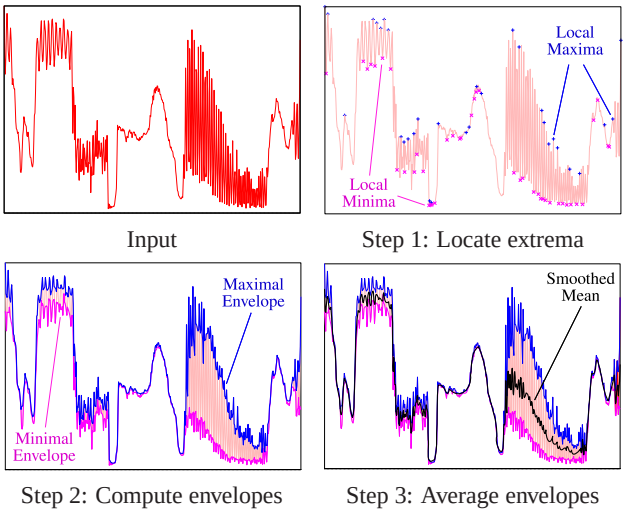
Extremal envelope construction: Given an image I and a set of pixels S (image local extrema), we compute an extremal envelope E using an interpolation technique that was proposed by Levin et al. [2004] for image colorization. In our context, we seek an interpolant E such that neighboring pixels $E(\mathbf{r})$ and $E(\mathbf{s})$ have similar values if $I(\mathbf{r})$ and $I(\mathbf{s})$ are similar. More formally, we minimize the functional

$$\sum_{\mathbf{r}} \left(E(\mathbf{r}) - \sum_{\mathbf{s} \in N(\mathbf{r})} w_{\mathbf{rs}} E(\mathbf{s}) \right)^2 \quad (1)$$



(a) Input I (b) Smoothed mean M_1 (c) Detail D_1 (d) Scanline-plots

Figure 6: *Plots showing the input intensities (red) along a row and its separation into detail (green) and mean (blue) by our algorithm. Despite the large amplitude of some oscillations they are extracted as detail D_1, while single edges of lower amplitude are preserved in the smoothed mean M_1.*



Input Step 1: Locate extrema

Step 2: Compute envelopes Step 3: Average envelopes

Figure 5: *The three steps of our smoothing algorithm illustrated with plots of intensity along the row shown in Figure 6.* Step 1: *We locate the local minima and maxima of the input (red). Note: The plot is along a row in the 2D input and extrema corresponding to some peaks seem to be missing since they lie on adjacent scanlines.* Step 2: *We compute the minimal (magenta) and maximal (blue) envelopes as edge-preserving interpolants through the minima and maxima respectively.* Step 3: *The smoothed mean (black) is computed as the average of the two envelopes.*

subject to the constraint

$$\forall\, \mathbf{p} \in S \quad E(\mathbf{p}) = I(\mathbf{p}).$$

$N(\mathbf{r})$ denotes the neighbors of $\mathbf{r}$, and weights

$$w_{\mathbf{rs}} \propto \exp\left(-\frac{(I(\mathbf{r}) - I(\mathbf{s}))^2}{2\sigma_r^2}\right) \tag{2}$$

are computed using the local variance σ_r^2 around $\mathbf{r}$. We adopt the approach of Levin et al. [2004] and minimize the quadratic functional using their weighted least squares formulation, which reduces to solving a sparse linear system with $N(\mathbf{r})$ defined as a 3×3 local neighborhood.

Smoothed mean: Performing the envelope construction independently on the minima and maxima of the image yields the minimal and maximal envelopes respectively. The smoothed mean image is computed as the average of these two envelopes (see Fig 5).

2.2 Multiscale decomposition

A single smoothing operation of I yields a detail image, D_1, that contains the finest-scale local oscillations and a mean, M_1, that represents a coarser trend. We obtain a multiscale decomposition of the input image by recursively extracting a number of detail layers from the mean. After n recursive smoothing operations, we obtain detail images $D_1, D_2, ..., D_n$ at increasing scales of coarseness and a residual mean image:

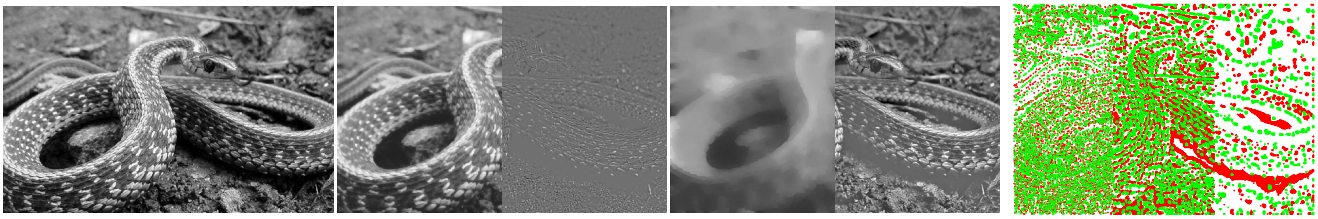
$$I(\mathbf{p}) = \sum_{i=0}^{n} D_i(\mathbf{p}) + M_n(\mathbf{p}). \tag{3}$$

Choosing $k = 3$ as the size of the extrema-location kernel (see Sec. 2.1) for the first smoothing step of I results in a detail D_1 that captures oscillations of frequency up to $3/2\,\text{pixel}^{-1}$. By increasing k, we effectively capture coarser oscillations while recursively smoothing M_1. Progressively increasing k through each recursive smoothing causes the different detail layers to contain increasingly coarse oscillations. In our experiments we found that the algorithm was not sensitive to the factor by which k was increased. For all the results in the paper we increased k by a constant value of eight, between iterations. Figure 7(d) visualizes the extrema of I, M_1 and M_2. For compact visualization, the three sets of extrema are shown in different vertical regions of the image.

2.3 Discussion

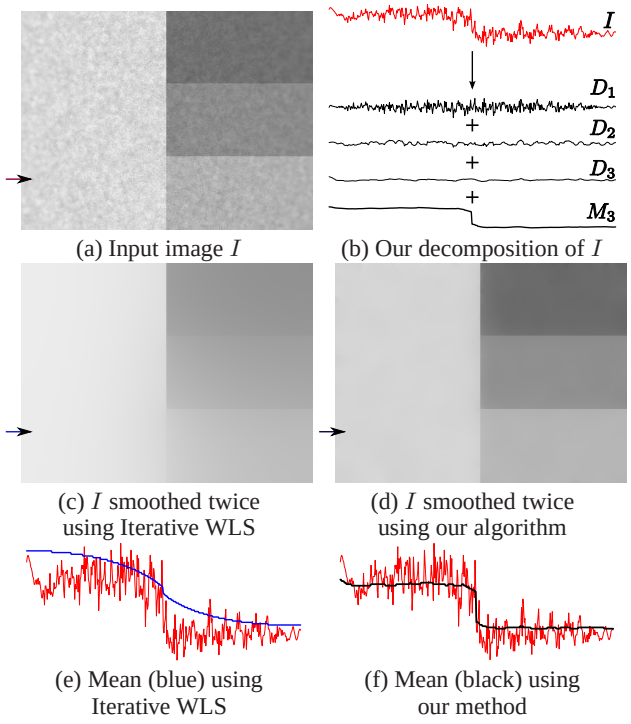
Effects of noise: For noisy input images, our algorithm effectively separates the noise if the scale of the noise does not match the scale of features in the input image. We repeated an experiment performed by Farbman et al. [2008], on a greyscale image with several step-edges of varying magnitude that was polluted with noise at two scales. Our decomposition algorithm effectively recovers the noise at different scales (see Fig. 8).

Edge preservation: Current edge-preserving image decompositions use local contrast to define edges. On the other hand, we define edges as regions where the *variation in the values of the neighboring extrema* is large. Our smoothing filter preserves edges because the extremal envelopes implicitly maintain fidelity to the data at pixels where the variation in the range values of the nearby extrema is large. Regions with large-amplitude, oscillations are smoothed effectively since the local extrema have similar values.



(a) Input image | (b) Base/detail after one smoothing operation | (c) Base/detail after two smoothing operations | (d) Extrema of input and two base layers (b) and (c)

Figure 7: *Our multiscale decomposition extracts features based on their spatial scale. An input image is shown along with its three-layer decomposition. The local extrema of the input image, the base layer in (b) and the base layer in (c) are shown as three abutting vertical regions in (d).*



(a) Input image I | (b) Our decomposition of I

(c) I smoothed twice using Iterative WLS | (d) I smoothed twice using our algorithm

(e) Mean (blue) using Iterative WLS | (f) Mean (black) using our method

Figure 8: *Results of applying our algorithm on a noisy image (courtesy of Farbman et al. [2008]). (a) The input image I is a piecewise constant image containing several step-edges of different magnitudes, to which noise was added at different scales. Our smoothing algorithm produces a better estimate of the mean while effectively extracting detail at multiple scales. (b) The result of our decomposition on a single row. (c) The result of smoothing I using iterative WLS [Farbman et al. 2008]. (d) The result of smoothing I using our algorithm. (e) A plot of the smoothed result (blue) using WLS filtering, along with the input (red). (f) A plot of the our smoothed result (black) with the input (red).*

Robustness to image scaling: Performing the decomposition of a scaled version of an image provides consistent results if the window used for extrema detection is scaled accordingly. The size of the kernel used in our extrema detection determines the largest frequency of oscillations that can be extracted as detail. To maintain consistency between decompositions of scaled versions of the input image it suffices to simply scale the the kernels by the same factor.

Sparse extrema: When the density of local extrema is very low, the interpolation [Levin et al. 2004] can become unstable. However, a low extremal density indicates that the underlying function is very smooth. Introducing artificial interpolation constraints (extrema) in smooth regions makes the interpolation stable. In practice, we insert artificial extrema in regions of the image that contain no extrema and are larger than a given threshold size (50×50 pixels).

Smoothing by contrast reduction: In traditional empirical mode decomposition [Huang 1998] of smooth 1D data, smooth interpolation schemes are used to construct the extremal envelopes. We use an edge-preserving interpolation scheme so that the smoothed mean preserves isolated discontinuities. The tendency of the interpolant to preserve large gradients may result in incomplete smoothing of oscillations in a single iteration. However, a combination of increasing the window size for extrema-location and performing the decomposition in the log-domain make this effect almost imperceivable. Another solution is to repeat each smoothing step (keeping k fixed) until the detail is completely extracted.

Features at boundaries of textured regions: large-amplitude oscillations that occur at the boundaries of textured regions are indistinguishable from edges. Figure 9 illustrates an example where, despite the high contrast, the spotted pattern on the hat is smoothed effectively while subtler shading is preserved on the coarse scale. However, the bright spots at the boundary with the ribbon are mistaken to be part of the ribbon. Handling such cases would require semantic information such as from an explicit pattern matching algorithm.

3 Results

We tested our smoothing and decomposition algorithms on a variety of images. On average, a four-layer decomposition of 1024×768 images took about 30 seconds using a naïve solver for computing the extremal envelopes. Using a simple multigrid solver, we were able to achieve a speedup of about 1.5. To locate extrema, we use a 3×3 kernel for the finest detail and progressively enlarge the kernel by a constant value (8) through the recursion for coarser layers.

3.1 Comparison

We wish to stress the difference in philosophies between current algorithms and our approach. Our novel definition of detail, as repetitive oscillatory features between local extrema, produces fundamentally different decompositions from existing solutions that interpret large gradients as edges to be preserved. The differences are primarily with coarse-scale features that have low contrast and fine-scale features that are highly contrasted.

Techniques, that extract low contrast features as detail, typically

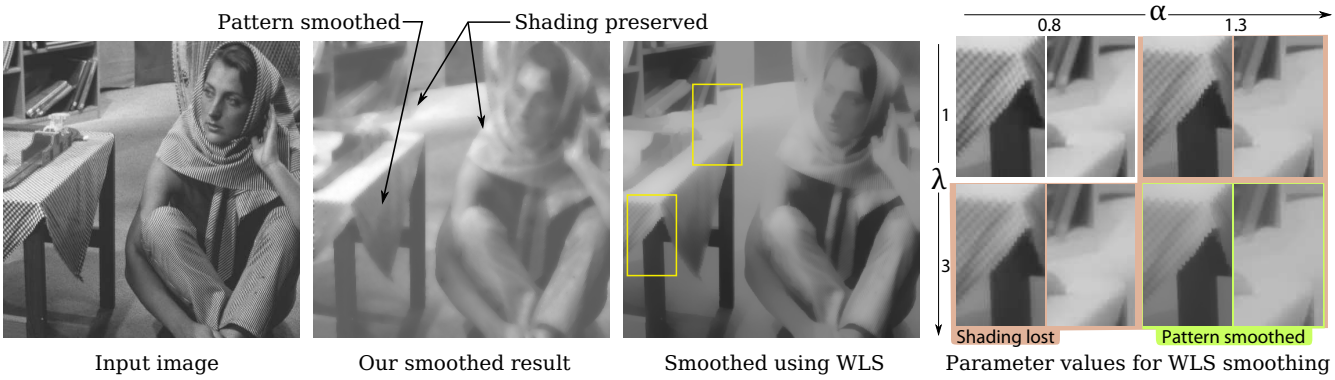


Figure 10: *The Barbara input image along with results of smoothing with the WLS filter [Farbman et al. 2008] using various combinations of the input parameters. Zooming into insets with contrasted texture and subtle shading, we see that gradient-based techniques are unable to preserve subtle, coarse features while smoothing fine, well-contrasted texture. Our method preserves subtle shading and effectively smoothes the texture.*

(a) Input image (b) Smoothed image

Figure 9: *Failure case: Although the high-contrast, spotted pattern on the hat is smoothed effectively while retaining subtler shading information, parts of this pattern on the boundary with the ribbon are indistinguishable from the ribbon. Although our definition of detail does not inherently disambiguate edges from partial oscillations of similar amplitude at boundaries, this is an extreme example. Handling such cases would require semantic information such as from an explicit pattern matching algorithm.*

demonstrate their utility using images where the low contrast detail also tends to be fine-scale. On such images, despite the difference in philosophies, our results are quite similar since fine-scale features extracted by our technique as detail also happen to be of low contrast. For example, using the flower example of Farbman et al. [Farbman et al. 2008] we achieve similar results (Fig. 11) since the details on the flower petals are fine-scale and of lower contrast than at the boundaries. In this paper, we focus on cases that produce different decompositions from gradient-based approaches.

Figure 15 compares the results of our technique with existing decomposition schemes. One key difference is that our decomposition extracts, earlier, fine-scale features (such as the pebbles towards the bottom of the image) as detail, while existing schemes extract low-contrast features (such as the large clouds) earlier as detail.

Figure 10 shows an example where the input contains texture that is more contrasted than some edges. Using a purely gradient dependent approach, smoothing the oscillation necessarily smoothes low-contrast edges (see also Fig. 3). Also, current decompositions can involve non-intuitive manipulation of input parameters across different images. In comparison, our technique is simple, smoothes texture, respects soft, single edges, preserves subtle shading and consistently smoothes a variety of images with widely different contrasts.

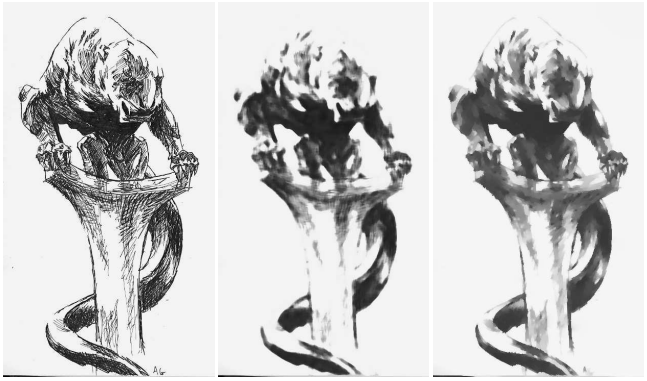
3.2 Applications

Multiscale decompositions of images, into layers of varying contrast, have been used in several applications including equalization and image abstraction [Farbman et al. 2008; Lischinski et al. 2006; Fattal et al. 2007]. In addition to these, we present applications that exploit a key property of our decomposition— the extracted layers correspond to superposed oscillations of increasing coarseness. We apply our decomposition to enhance detail (image equalization) and to remove detail (estimating tone from cross-hatched images, separating texture from illumination, illumination transfer).

Hatch to tone: Few techniques are able to recover tone from images with hatching or stippling, while preserving edges. The difficulty lies in retaining edges depicted by these techniques while smoothing high-contrast variation. Smoothing filters like the bilateral filter or weighted least squares filter are not very useful in this context. Figure 12 shows the residual from running three iterations of our smoothing algorithm on a cross-hatched input image. We smooth fine-scale oscillations, ideally to their flat means, earlier in the process. However, in the case of non-homogeneous, high-contrast oscillations, the edge preserving nature of the non-linear extremal interpolants causes the contrast of the oscillations to be reduced considerably but not completely. Consequently the computed mean tends to contain residual oscillations that are grayscale. The amplitude of these residual oscillations depends on its original wavelength; fine oscillations leave weaker residuals than coarse ones. Over multiple iterations of such smoothing applied on binary (or highly contrasted) hatched images, the complex interplay between homogeneity of oscillations in 2D and grayscale residuals from previous iterations tends to result in a smoothed image where the tone at each pixel is directly related to the frequency of local oscillations. While we smooth variation, the edges of variations are well preserved. We compare our solution with a median filter. The problem with the latter is that, using a small kernel size, tone is not recovered at a coarse scale and Increasing the kernel size wipes out thin features like outlines. Another drawback of the median filter is

Figure 11: *Fine-scale enhancement of the input image (left) using WLS [Farbman et al. 2008] (middle) and our technique (right) provides similar results with subtle differences since the detail on the petals are of, both, low contrast and fine scale. The WLS method fails to enhance fine-scale detail that are high-contrast such as the serrations on the leaves in the background and specularities on the small leaves on the right. In addition, coarser features such as the subtle discoloration on the defocused top-left portion of the image are enhanced as detail by the WLS method simply because they are low contrast.*

that the filter only selects pixel levels that are present in the input image.

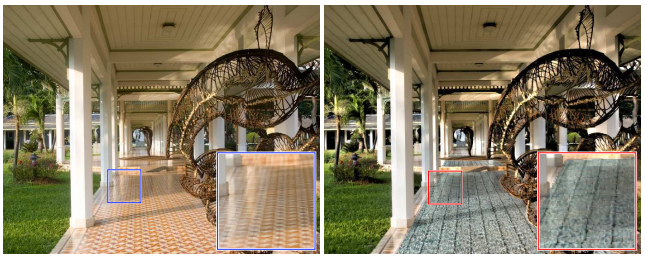


(a) Input (b) Median filtering (c) Our method

Figure 12: *(b) Applying a median filter has two disadvantages: Choosing a large kernel size washes out thin edges while choosing a small kernel size does not smooth the hatched pattern; also the median filter simply selects one of the existing grey levels and cannot produce intermediate shades of grey. (c) The residual after three iterations of smoothing using our algorithm yields a good estimate of the tone while preserving the edges of hatched regions.*

Separating fine texture and coarse shading: We are able to separate fine texture from shading, provided the oscillations of the texture and shading are of different scales. Although we make the same assumption as Oh et al. [2001] that illumination information is "lower frequency" than texture, we do not make any assumptions on the contrast of the texture. Since Oh et al. use the bilateral filter, they are prone to the additional assumption that the contrast of the texture and shading are vastly different. We demonstrate the effectiveness of our algorithm by retexturing an image containing high-contrast texture, while retaining shading on the newly painted texture (see Fig. 13). We achieve this by transferring the coarsened luminance of the input image onto its edited version.

Image equalization: The layers from our decomposition can be seen as an adaptive basis that sum to the input. By considering different linear combination of these layers, we show that detail at different scales can be exaggerated. In practice, since we manipulate the log-luminance channel, we perform the linear combinations in log space. Current equalization techniques define detail as low contrast. Instead, we are able to control relative contrasts of features based on their scales (see Fig.1). More examples of image equalization are presented in the video.



(a) Input image (b) Illumination transfer onto painted texture

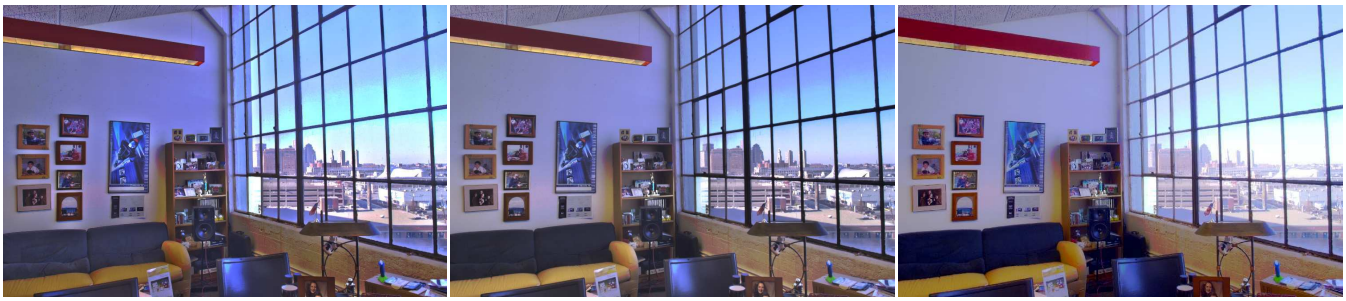
Figure 13: *Our edge preserving decomposition separates an input image into layers containing detail at different scales. (a) The tiled texture on the floor is finer than illumination effects such as glossy reflections and shadows. (b) The coarse illumination information is extracted from (a) and combined with the fine texture information extracted from (b) to preserve shadows and subtle effects such as glossy reflections of pillars on the newly painted texture (inset).*

High dynamic range (HDR) images: Although filters that extract detail based on contrast (WLS and bilateral filters) are more appropriate tools for tone-mapping, in practice, we find that our equalizations produce reasonable results (see Fig. 14). An advantage of our method is intuitive and consistent parameter values across different images. However, since we filter based on scale and not contrast, specialized techniques may be preferable for input where the HDR content is spread across significantly different spatial scales.

4 Conclusion

We have presented a novel definition for image detail as oscillations between local minima and maxima. While existing decomposition algorithms extract detail based on a notion of contrast, our definition of detail captures the scale of spatial oscillations, locally.

Building on our definition of detail, we proposed a simple algorithm to smooth an input image. By recursively performing the smoothing with extrema detection at multiple scales, we performed a decomposition of the input image into multiple-scale layers of detail and a coarse residual. Our algorithm smoothes high-contrast texture while preserving salient edges. Finally, we exploited this ability by applying our decomposition in a variety of applications.



(a) Tone-mapped using the bilateral filter (b) Tone-mapped using the WLS filter (c) Our equalized result

Figure 14: *Comparison of our equalization against tone-mapping operators on an example high dynamic range (HDR) input image. (a) and (b) have been directly taken from [Durand and Dorsey 2002] and [Farbman et al. 2008] respectively. (c) is obtained using our 2-layer equalization where the base layer is scaled to half and recombined with the detail. Although our notion of detail is based on spatial scale and not contrast, our equalization can be used to achieve basic tone-mapping by scaling down the layer(s) with HDR content.*

Acknowledgements

We thank Adrien Bousseau and Alexandrina Orzan for their help in creating the video. We also thank the MIT, ARTIS and SIGGRAPH reviewers for their insightful suggestions. This work was supported by funding from ANR 'HFIBMR' (ANR-07-BLAN-0331), INRIA Equipe Associée with MIT Flexible Rendering and the INRIA post-doctoral program.

References

BAE, S., PARIS, S., AND DURAND, F. 2006. Two-scale tone management for photographic look. *ACM Transactions on Graphics 25*, 3, 637–645.

BURT, P. J., AND ADELSON, E. H. 1983. The laplacian pyramid as a compact image code. *IEEE Trans. on Communications COM-31,4*, 532–540.

CHEN, J., PARIS, S., AND DURAND, F. 2007. Real-time edge-aware image processing with the bilateral grid. *ACM Transactions on Graphics*, 103.

CHOUDHURY, P., AND TUMBLIN, J. 2005. The trilateral filter for high contrast images and meshes. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Courses*, ACM, New York, NY, USA, 5.

DAMERVAL, C., MEIGNEN, S., AND PERRIER, V. 2005. A fast algorithm for bidimensional emd. *Signal Processing Letters, IEEE 12*, 10 (Oct.), 701–704.

DURAND, F., AND DORSEY, J. 2002. Fast bilateral filtering for the display of high-dynamic-range images. In *ACM Transactions on Graphics: SIGGRAPH '02*, ACM Press, New York, NY, USA, 257–266.

FARBMAN, Z., FATTAL, R., LISCHINSKI, D., AND SZELISKI, R. 2008. Edge-preserving decompositions for multi-scale tone and detail manipulation. *ACM Transactions on Graphics*, 67.

FATTAL, R., AGRAWALA, M., AND RUSINKIEWICZ, S. 2007. Multiscale shape and detail enhancement from multi-light image collections. *ACM Transactions on Graphics*, 51.

HUANG. 1998. The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 454*, 1971 (March), 903–995.

LAGENDIJK, R. L., BIEMOND, J., AND BOEKEE, D. E. 1988. Regularized iterative image restoration with ringing reduction. *IEEE Trans. on Signal Processing (Acoustics, Speech, and Signal Processing) 36*, 12, 1874–1888.

LEVIN, A., LISCHINSKI, D., AND WEISS, Y. 2004. Colorization using optimization. *ACM Transactions on Graphics 23*, 689–694.

LI, H., YANG, L., AND HUANG, D. 2005. The study of the intermittency test filtering character of hilbert-huang transform. *Mathematics and Computers in Simulation 70*, 1, 22–32.

LISCHINSKI, D., FARBMAN, Z., UYTTENDAELE, M., AND SZELISKI, R. 2006. Interactive local adjustment of tonal values. *ACM Transactions on Graphics 25*, 3, 646–653.

LIU, Z., AND PENG, S. 2005. Boundary processing of bidimensional emd using texture synthesis. *Signal Processing Letters, IEEE 12*, 1 (Jan.), 33–36.

NUNES, J., NIANG, O., BOUAOUNE, Y., DELECHELLE, E., AND BUNEL, P. 2003. Texture analysis based on the bidimensional empirical mode decomposition with gray-level co-occurrence models. *Signal Processing and Its Applications, 2003. Proceedings. 2* (July), 633–635 vol.2.

OH, B. M., CHEN, M., DORSEY, J., AND DURAND, F. 2001. Image-based modeling and photo editing. In *Proceedings of SIGGRAPH 2001*, ACM, NY, USA, 433–442.

PATTANAIK, S. N., FAIRCHILD, M., FERWERDA, J., AND GREENBERG, D. P., 1998. Multiscale model of adaptation, spatial vision and color appearance.

RAHMAN, Z. U., AND WOODELL, G. A. 1997. A multi-scale retinex for bridging the gap between color images and the human observation of scenes. In *IEEE Trans. on Image Processing: Special Issue on Color Processing 6(7*, 965–976.

SERRA, J., AND VINCENT, L. 1992. An overview of morphological filtering. In *Circuits, Systems and Signal Processing*, 47–108.

TOMASI, C., AND MANDUCHI, R. 1998. Bilateral filtering for gray and color images. In *In Proc. of the Sixth International Conference on Computer Vision, Bombay, India, January 1998.*

TUMBLIN, J., AND TURK, G. 1999. Lcis: a boundary hierarchy for detail-preserving contrast reduction. In *Proceedings of SIGGRAPH '99*, ACM Press/Addison-Wesley Publishing Co., NY, USA, 83–90.

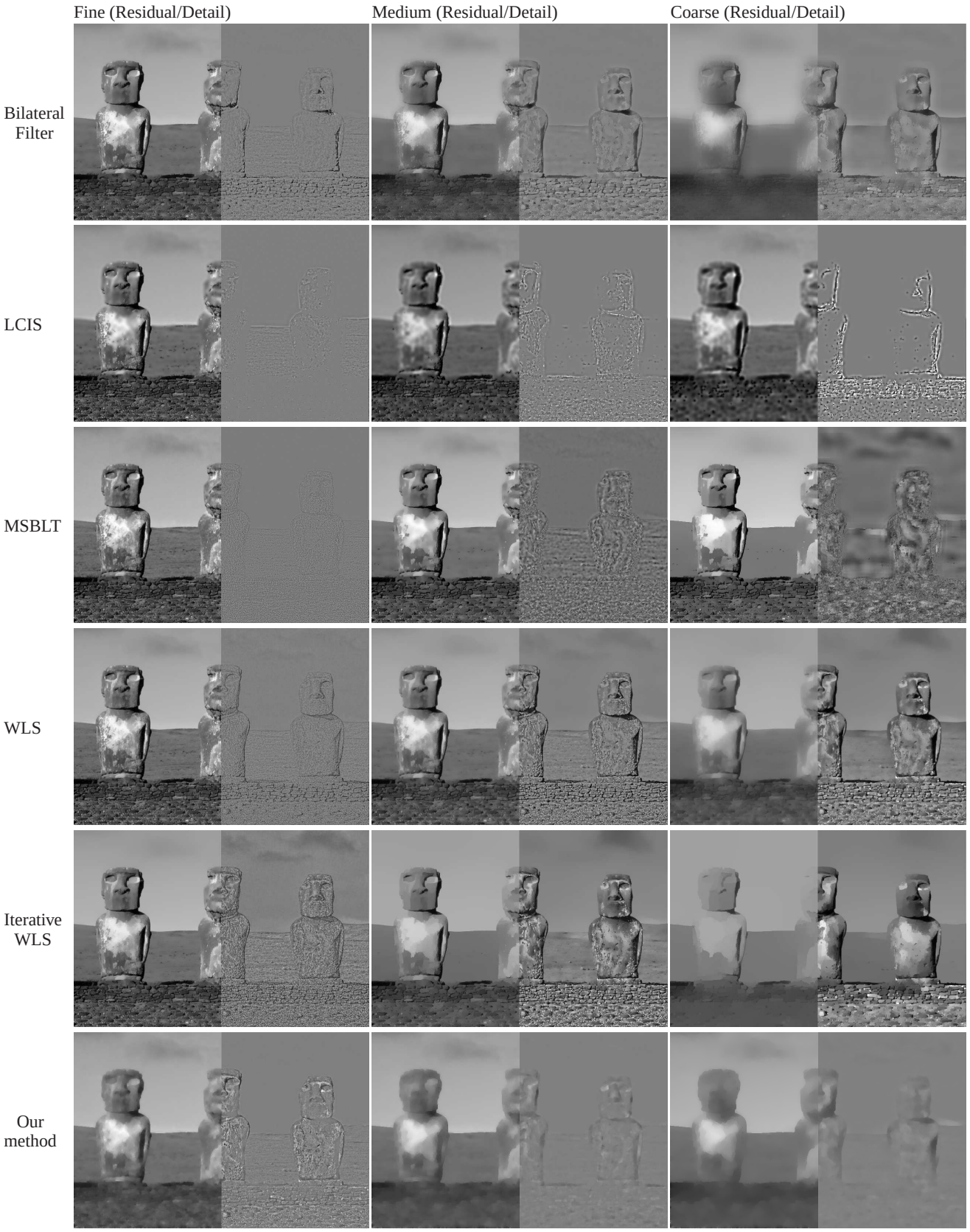


Figure 15: *Comparison of our results with existing approaches: Bilateral filtering [Chen et al. 2007], MSBLT [Fattal et al. 2007], LCIS [Tumblin and Turk 1999], WLS, iterative WLS [Farbman et al. 2008]. Our smoothing extracts features based on spatial scale while other methods smooth low-contrast features first. Using our decomposition, the pebbles and stones towards the bottom of the image are extracted as fine- and medium-scale detail respectively, even though they are well contrasted. On the other hand, despite their low contrast, the clouds are not extracted as detail due to their coarse scale. The comparison images have been directly taken from [Farbman et al. 2008].*

A Local Frequency Analysis of Light Scattering and Absorption

LAURENT BELCOUR
Manao†, Inria Bordeaux Sud-Ouest
and
KAVITA BALA
Cornell University
and
CYRIL SOLER
Maverick, Inria Rhône-Alpes

Rendering participating media requires significant computation, but the effect of volumetric scattering is often eventually smooth. This paper proposes an innovative analysis of absorption and scattering of local light fields in the Fourier domain, and derives the corresponding set of operators on the covariance matrix of the power spectrum of the light field. This analysis brings an efficient prediction tool for the behavior of light along a light path in participating media. We leverage this analysis to derive proper frequency prediction metrics in 3D by combining per-light path information in the volume.

We demonstrate the use of these metrics to significantly improve the convergence of a variety of existing methods for the simulation of multiple scattering in participating media. Firstly, we propose an efficient computation of second derivatives of the fluence, to be used in methods like irradiance caching. Secondly, we derive proper filters and adaptive sample densities for image-space adaptive sampling and reconstruction. Thirdly, we propose an adaptive sampling for the integration of scattered illumination to the camera. Finally, we improve the convergence of progressive photon beams by predicting where the radius of light gathering can stop decreasing. Light paths in participating media can be very complex. Our key contribution is to show that analyzing local light fields in the Fourier domain reveals the consistency of illumination in such media, and provides a set of simple and useful rules to be used to accelerate existing global illumination methods.

Categories and Subject Descriptors: I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Raytracing*

Additional Key Words and Phrases: global illumination, participating media, adaptive sampling, frequency analysis

†Inria - LP2N (CNRS, Univ. Bordeaux, IOGS) - LaBRI (CNRS, Univ. Bordeaux)

DOI 10.1145/XXXXXXX.YYYYYYY
http://doi.acm.org/10.1145/XXXXXXX.YYYYYYY

1. INTRODUCTION

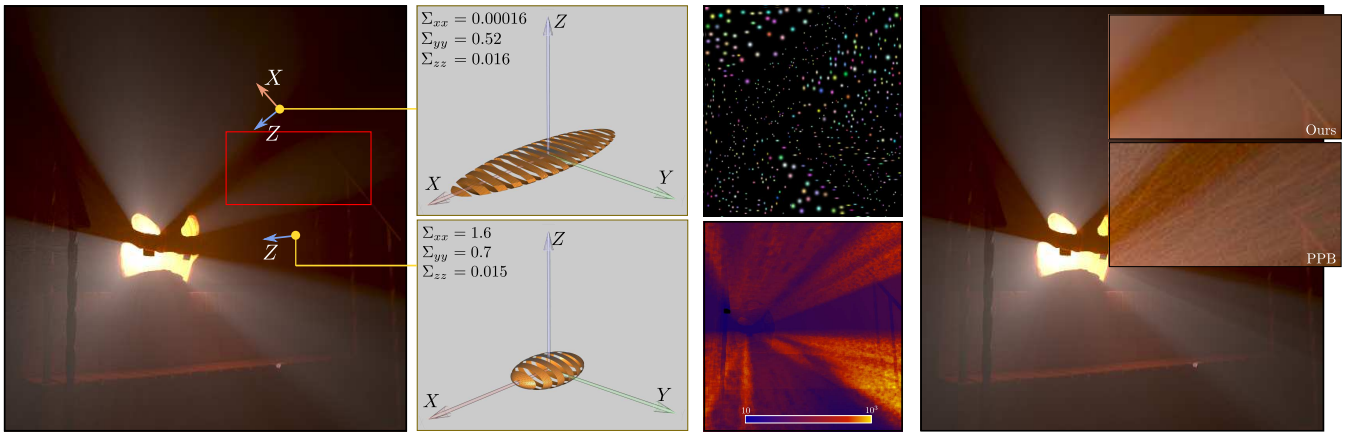
Rendering participating media is challenging because of the high cost of simulating scattering events. But participating media mostly blur out details, and decrease the contrast of images: some image regions appear almost locally constant, and light beams are practically constant in the direction of light propagation. In this paper we introduce a new frequency analysis of local light fields in participating media. We show the effect that volumetric scattering has on lowering frequency content and contrast. We derive the associated theoretical framework and provide tools to optimize participating media rendering algorithms using the frequency content of light transport.

Scattering is a long standing problem in computer graphics where a range of techniques, with varying trade-offs between performance and accuracy, have been proposed to simulate the interaction of light with participating media. Unbiased methods such as path tracing [Lafortune and Willems 1993] and Metropolis light transport [Veach and Guibas 1997] provide accuracy, but often at a prohibitive cost. Photon mapping based approaches handle participating media [Jensen and Christensen 1998; Knaus and Zwicker 2011] with different trade-offs. Methods such as Photon Beams [Jarosz et al. 2011] efficiently simulate low order scattering, relying on the accumulation of illumination primitives (e.g., points or beams) to compute images. While some approaches exploit the lower frequency nature of lighting in participating media, to our knowledge, there is no existing literature on *a priori* frequency analysis of local light fields in volume transport.

For non-volumetric surface-based rendering, Durand et al. [2005] introduced a frequency analysis of light transport. We extend this framework to characterize the behavior, in the Fourier domain, of light traveling and scattering inside participating media. Methods exist that use the Fourier transform as a global transform operator in 3D to decouple the frequencies in the scattering equation [Ishimaru 1997]. Instead, our extension to the frequency analysis framework applies to 4D local light fields, along light paths in the medium.

We build on covariance analysis [Belcour et al. 2013], an efficient and practical representation of the covariance matrix of the frequency spectrum of the local light field. It was used to accelerate the rendering of motion and defocus blur. The covariance matrix representation conveys the required information on the Fourier transform of the light field, at a very small cost, making it tractable for path-tracing.

In this paper, we extend the covariance representation to global illumination in participating media, including multiple scattering. We show that our new formulae for participating media fit nicely in



(a) Our frequency estimate combined with Progressive Photon Beams (25 min)

(b) Predicted 3D covariance of fluence spectrum

(c) Image space filters & Camera rays sample density

(d) Reference (Prog. Photon Beams, 4h44min) Inset: Equal time comparison (25min)

Fig. 1. We propose a frequency analysis of light transport in participating media, a broad theoretical tool that allows improvements in a wide variety of algorithms. From the predicted 3D covariance of the fluence in the Fourier domain (b), we derive three different sampling metrics in the 3D volume. We present multiple example uses of these metrics: to improve image-space adaptive sampling density and reconstruction, we provide sampling density and reconstruction filters (c-top); to improve light integration along camera rays, we evaluate a required number of samples along those rays (c-bottom); to improve progressive photon beams, we derive the optimal reconstruction radius based on the frequency content (a and d). In order to ease comparisons, we scaled the covariance graphs (b), and increased the luminosity of insets (d). This scene is composed of a pumpkin modeled by user Mundomupa of blendswap.com provided under creative common license CC-BY, and of a house front modeled by Jeremy Birn.

the existing framework when the medium is not optically thick (as in subsurface scattering). We use the covariance information of the local light field spectrum in 4D along light paths to predict the 3D covariance of the windowed spectrum of fluence in volumes with participating media. We propose four application scenarios where this quantity proves useful (see Figure 1). The contributions of this paper are:

—A local analysis of scattering and absorption in the Fourier domain along a light path, in heterogeneous participating media. The model is compatible with multiple scattering.

—A compact representation of attenuation and scattering in the Fourier domain, using covariance matrices.

—The combination of covariance from many light paths in the medium into usable sampling metrics in 3D.

—Four different computation scenarios that benefit from our analysis: computation of second derivatives of the fluence; image space adaptive sampling and reconstruction; adaptive sampling of scattered illumination along rays from the camera; and progressive photon beams.

Note that the term “spectral analysis” usually has multiple meanings; it is sometimes used to refer to the eigenanalysis of linear operators. In this paper the term spectrum refers to the frequency spectrum of the Fourier transform of light fields.

2. PREVIOUS WORK

We categorize related work into research on the frequency analysis of light transport, and on the volume rendering of participating media.

2.1 Fourier domain methods for scattering.

In these methods, the Fourier transform is used as a tool for solving the scattering equation at once in the entire domain [Duderstadt and Martin 1979; Ishimaru 1997]. Some methods use a different basis for certain dimensions, such as the Chebychev basis [Kim and Moscoso 2003], or spherical harmonics [Dave 1970]. These methods in general depend on a combination of very specific constraints: infinite or spherical domains [Dave and Gazdag 1970], periodic boundary conditions [Ritchie et al. 1997], isotropic scattering functions [Rybicki 1971], and mostly homogeneous scattering functions. These conditions make such methods not very suitable to computer generated images where the constraints of uniformity and periodicity can hardly be satisfied.

Our approach is fundamentally different: we use the Fourier transform as a local tool in the 4D ray space to predict bandwidth—as opposed to globally solving the equations—which allows us to handle non homogeneous participating media.

2.2 Volume rendering.

The field of rendering participating media has a long history. Volume rendering based on ray tracing techniques was first proposed for forward path tracing integration [Kajiya and Von Herzen 1984]. It has been expanded afterwards to other integration schemes: Lafortune and Willems [1996] extended bidirectional path tracing; Pauly et al. [2000] adapted Metropolis for participating media. Photon mapping [Jensen and Christensen 1998] has been shown to be efficient in generating high frequency light patterns such as caustics. Cerezo et al. [2005] surveys the state-of-the-art, though it is a bit dated.

Recently, various extensions to photon mapping and progressive photon mapping use photon beams, rather than point sampling along rays, to greatly improve the performance of volume rendering [Jarosz et al. 2008; Jarosz et al. 2011; Jarosz et al. 2011; Knaus

and Zwicker 2011]. These methods however, remain unaware of image complexity, and rely on an accumulation of illumination primitives (e.g., points or beams) to compute an image that will eventually be very smooth.

Several virtual point light (VPL)-based algorithms for volumetric rendering trade-off quality and performance. Light cuts and variants [Walter et al. 2005; Walter et al. 2006; Walter et al. 2012] achieve scalable rendering of complex lighting with many VPLs for motion blur, depth of field, and volumetric media (including for oriented media [Jakob et al. 2010]). These scalable approaches couple error bounded approximations with simple perceptual metrics. For interactive VPL rendering of participating media, Engelhardt et al. [2010] introduce a GPU-friendly bias compensation algorithm. Novak et al. [2012] spread the energy of virtual lights along both light and camera rays, significantly diminishing noise caused by singularities.

Multiple approaches aim at efficiently computing low-order scattering in refractive media. Walter et al. [2009] compute single scattering in refractive homogeneous media with triangle boundaries. Ihrke et al. [2007] solve the eikonal equation with wavefront tracing for inhomogeneous media with varying refractive indices and Sun et al. [2010] develop a line gathering algorithm to integrate complex multiple reflection/refraction and single scattering volumetric effects for homogeneous media.

2.3 Efficient sampling and reconstruction methods

Some works perform adaptive sampling or local filtering using heuristics based on the frequency of light transport, without explicitly computing frequency information. Adaptive sampling for single scattering [Engelhardt and Dachsbacher 2010] permits resampling when detecting an occlusion. This approach finds epipolar lines, sparsely samples and interpolates along these lines, but finds occlusion boundaries to preserve high frequency details. An epipolar coordinate system [Baran et al. 2010; Chen et al. 2011] allows to interactively compute single scattering in volumetric media by exploiting the regularity of the visibility function.

The structure of the light field [Levoy and Hanrahan 1986; Gortler et al. 1986] can be exploited to perform adaptive sampling or reconstruction. For surface radiance computation, Lehtinen et al. [2011] exploits anisotropy in the temporal light field to efficiently reuse samples between pixels, and perform visibility-aware anisotropic reconstruction to indirect illumination, ambient occlusion and glossy reflections. Ramamoorthi et al. [2012] derived a theory of Monte Carlo visibility sampling to decide on the best sampling strategies depending on a particular geometric configuration. Mehta et al. [2012] derives the sampling rates and filter sizes to reconstruct soft shadows from a theoretical analysis to consider axis-aligned filtering.

Irradiance caching methods [Jarosz et al. 2008] inherently perform filtering in the space of the irradiance by looking at the irradiance gradient. For example, Ribardiere et al. [2011] perform adaptive irradiance caching for volumetric rendering. They predict variations of the irradiance and map an ellipsoid to define the non-variation zone with respect to a local frame.

2.4 Frequency analysis of light transport.

In their frequency analysis of light transport, Durand et al. [2005] studied the frequency response of the radiance function to various radiative transport phenomena (such as transport, occlusion and reflection). Other works on this subject [Egan et al. 2009; Soler et al. 2009; Belcour and Soler 2011; Bagher et al. 2012] have enriched the number of effects to be studied (motion, lens) and showed that filtering and adaptive sampling methods can benefit from frequency analysis. Yet, some radiative phenomena have not been studied in this framework, including refraction and scattering. We aim to fill a part of this gap by bringing comprehension of the frequency equivalent of volume scattering and attenuation operators, and showing the usefulness of such analysis with a few practical applications.

A frequency analysis has been carried out for shadows specifically by Egan et al.in 4D to build sheared reconstruction filters for complex visibility situations [Egan et al. 2011], or in the case of occlusion by distant illumination [Egan et al. 2011].

3. BACKGROUND: COVARIANCE OF LOCAL SPECTRUM

Our ultimate goal is to provide a general, efficient tool for predicting the variations of the illumination, at different stages of the calculation of global illumination, so as to make sampling and reconstruction methods the most efficient possible. In prior work [Belcour et al. 2013], it was demonstrated that the covariance of the spectrum of the local light field along rays does this job. In this paper, we perform the mathematical analysis to extend this approach to multiple scattering in participating media. This section recalls the basics about local light fields, Fourier analysis of light transport and the covariance representation of the spectrum as background.

3.1 Local light fields

We call the *local light field* the 4D field of radiance in the 4D neighborhood of a ray. Our space of study is the 4D domain of tangential positions around a central ray [Igehy 1999; Wand and Straßer 2003]. It is parameterized by two spatial and two angular coordinates, defined with respect to the plane orthogonal to the ray at a 3D position $\mathbf{x}$ (See Figure 2).

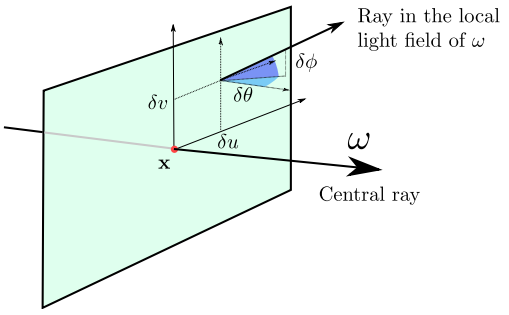


Fig. 2. Parameterization of a local radiance light field around a ray of direction ω. We use δu, δv as the transverse spatial coordinates of the ray and $\delta\theta$, $\delta\phi$ as its angular coordinates.

3.2 Fourier analysis

Durand et al. analyzed the various effects a local light field undergoes along a light path [Durand et al. 2005]. They showed that the effect of light transport operators such as reflection, free space transport, and occlusion, all correspond to simple operators on the Fourier spectrum of the light field. These operators and their equivalent operator in the Fourier domain are listed in Table I.

Space travel	Occlusion	BRDF	Rotation by angle φ	Scale $f(\lambda_1 x_1, ..., \lambda_4 x_4)$
$\Sigma' = T_d^T \Sigma T_d$	$\Sigma' = \Sigma + O$	$\Sigma' = \left(\Sigma^{-1} + B\right)^{-1}$	$\Sigma' = R_\varphi^T \Sigma R_\varphi$	$\Sigma' = \Lambda \Sigma \Lambda$
$T_d = \begin{bmatrix} 1 & 0 & -d & 0 \\ 0 & 1 & 0 & -d \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$O = \begin{bmatrix} O_{xx} & O_{yx} & 0 & 0 \\ O_{xy} & O_{yy} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$	$B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & B_{\theta\phi}^{-1} & \\ 0 & 0 & & \end{bmatrix}$	$R_\varphi = \begin{bmatrix} cos(\varphi) & -sin(\varphi) & 0 & 0 \\ sin(\varphi) & cos(\varphi) & 0 & 0 \\ 0 & 0 & cos(\varphi) & -sin(\varphi) \\ 0 & 0 & sin(\varphi) & cos(\varphi) \end{bmatrix}$	$\Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & \lambda_3 & 0 \\ 0 & 0 & 0 & \lambda_4 \end{bmatrix}$

Fig. 3. Operations on the light field and their equivalence on the *covariance matrix of the light field's spectra* at various stages of light transport (see Belcour et al. [2013] for detailed derivations). The middle row shows each operator applied to the covariance Σ while the bottom row details the constant matrices involved. Occlusion adds spatial covariance (O is the 2D spatial covariance of the occluder), BRDF removes angular covariance ($B_{\theta\phi}$ is the angular covariance of the BRDF locally to the reflected direction), while rotation and free-space travel only cause a re-parameterization.

Table I. Operators on the light field function along a light path, and their Fourier equivalent.

Effect	**Light space operator**	**Fourier operator**
Space travel	Angles→space shear	Space→angles shear
Occlusion	Product with visibility	Convolution by occluder's spectrum
Projection	Scale	Inverse scale
BRDF integral	Convolution by BRDF	Product with BRDF's spectrum

3.3 Covariance representation

It is not practical to perform a complete calculation of the full spectrum, especially on a per light path basis. Fortunately, we do not need the full spectrum of local light fields to perform useful predictions about how the local light field behaves. The relevant information needed is how far and in which directions the spectrum spreads in the Fourier domain. It was demonstrated that the covariance matrix of the power spectrum of the local light field is sufficient to maintain this information [Belcour et al. 2013].

In the most general case, assuming non-static scenes, local light fields are defined over space, angle, and time, making the light field and its power spectrum a 5D function. The effect of motion is derived independently of the other 4D operators using a change of coordinates in time [Belcour et al. 2013]. In the present document, we chose to focus on static scenes only, and work with 4D covariance matrices.

For any zero-centered, non-negative real function f defined over the 4D domain, the covariance matrix of f is a 4×4 matrix, denoted as Σ, and defined by:

$$\forall (i,j) \in \{1,...,4\}^2 \quad \Sigma_{i,j} = \frac{1}{\int_\Omega f} \int_{\mathbf{x}\in\Omega} (\mathbf{x}.\mathbf{e}_i)(\mathbf{x}.\mathbf{e}_j) f(\mathbf{x}) d\mathbf{x} \quad (1)$$

In this equation, $\mathbf{e}_i$ is the i^{th} vector of the canonical basis of the 4D space Ω, while $(\mathbf{x}.\mathbf{y})$ is the dot product of vectors $\mathbf{x}$ and $\mathbf{y}$.

The covariance matrix has very interesting properties in terms of what we actually need:

—its eigenvectors indicate in which direction function f spreads the most and where it spreads the least;

—its eigenvalues are the variance of the function in all 4 principal directions;

—it is additive, which allows us to accumulate the covariance matrices of rays. More specifically, the Monte Carlo estimate of a composed covariance matrix of many rays is the weighted average of the individual covariance matrices with radiance as weights.

Therefore, the covariance of the *power spectrum* (amplitude of the spectrum) of the local light field can provide us information about sampling and integrating the light field function [Belcour et al. 2013]. In the remaining text, we use the term *spectral covariance* to mean the covariance of the power spectrum of a given quantity.

3.4 Relationship with a Gaussian approximation

When the covariance Σ is non-degenerate, it also coincides with the covariance matrix of the Gaussian g defined by

$$g(\mathbf{x}) = e^{-\mathbf{x}^T \Sigma^{-1} \mathbf{x}}$$

Therefore, representing a function by its covariance matrix Σ is just as good as approximating that function by the Gaussian above. Such an approximation appears very relevant for power spectra, which—just like Gaussians—are zero-centered radially symmetric functions.

However, in order to handle degenerate cases, using a covariance matrix is more practical than handling Gaussian functions. Besides, all transformations we perform over the local light field directly turn into algebraic operations on the covariance matrix, with no further approximation except for multiplication [Belcour et al. 2013].

3.5 Covariance algebra

All operators listed in Table I directly act on the covariance matrix as algebraic operations, most of which are left-and-right products with constant matrices (see Figure 3). Obviously missing in this list are the operators to model the effect of volumetric attenuation and scattering over the local light field along a light path. We derive them in the next section. We also need an efficient way of combining the 4D light path covariance information in the 3D domain, to predict how smooth the diffused illumination will eventually be, and where and how to sample it. This is done in Section 5.

3.6 Relationship with second derivatives

We show in Appendix A, that for any function f with good regularity in the neighborhood of a given point $\mathbf{x}_0$, the covariance matrix *of the windowed spectrum* of f in the neighborhood of $\mathbf{x}_0$ corresponds to the Hessian matrix of f *in the primal domain* at $\mathbf{x}_0$, in the coordinate system of the principal curvatures:

$$\forall i \;\; \Sigma_{ii}(\hat{f}) = \frac{1}{4\pi^2 |f(\mathbf{x}_0)|} \left| \frac{\partial^2 f}{\partial x_i^2}(\mathbf{x}_0) \right| \quad (2)$$

This last property will be very useful later on in our analysis, as a practical method to compute covariance matrices of the windowed spectrum of some signals (e.g. the phase functions) around particular points.

4. FOURIER ANALYSIS FOR PARTICIPATING MEDIA

In this section, we extend the frequency analysis of light transport to participating media. We follow a light path, bouncing possibly multiple times, into the medium. Our model is therefore compatible with multiple scattering. We derive the equations to model the effect of two operators over the frequency spectrum of the local light field around this light path: absorption and scattering. We first perform a first order expansion of the phenomena. We then study the Fourier equivalent of the two operators, and show how to express them using algebraic operations on the spectral covariance matrices of the light field. Table II summarizes our notations.

Table II. Definitions and notations used in the paper.

$\delta u, \delta v, \delta\theta, \delta\phi$	Spatial and angular local coordinates
$l(\delta u, \delta v, \delta\theta, \delta\phi)$	local light field function
$\Omega_u, \Omega_v, \Omega_\theta, \Omega_\phi$	Spatial and angular variables in Fourier space
$\hat{l}(\Omega_u, \Omega_v, \Omega_\theta, \Omega_\phi)$	Fourier spectrum of the local light field
Σ	4D covariance of the local light field spectrum
Γ	3D covariance of windowed spectrum of fluence
δt	Coordinate along the central direction of travel
$\kappa(x, y, z)$	Volumetric absorption at 3D position (x, y, z)
$\kappa_{uv}(u, v)$	Volumetric absorption in plane orthogonal to a ray
$\omega, \omega_i, \omega_s$	directions of light (general, incident, scattered)
$\rho(\omega_i, \omega_s)$	phase function for directions ω_i, ω_s
$\overline{\rho}(\delta\theta, \delta\phi)$	phase function around ω_i, ω_s, *i.e.* $\overline{\rho}(0,0) = \rho(\omega_i, \omega_s)$
ρ_g	Henyey-Greenstein function with parameter g
α	Finite angle for scattered direction
$\otimes_{\Omega_u\Omega_v}$	Convolution operator in the Fourier Ω_u, Ω_v plane

Although the behavior of individual light paths in participating media is potentially complicated, we will show that in the Fourier domain, absorption acts like visibility, and scattering acts like reflectance. Not only does this fit elegantly into the existing framework, but it also results in a very simple frequency prediction tool for efficiently rendering participating media.

4.1 Volumetric absorption

We first consider the effect of volumetric absorption. When the density of particles is not constant in space, energy is not uniformly absorbed as light travels through the medium. This creates an increase in spatial frequencies in the signal (similar to shadows), which further propagates to the angular domain because of the travel of light. We study the effect of volumetric absorption by a density function $\kappa(x, y, z)$ acting as an extinction coefficient along a ray, for a small travel distance δt along ω (see Figure 4).

The attenuated light obeys the following differential equation [Cerezo et al. 2005]:

$$\frac{\partial(l(\mathbf{x} + t\omega, \omega))}{\partial t}(0) = -\kappa(\mathbf{x}) l(\mathbf{x}, \omega) \tag{3}$$

We perform a first order approximation of the absorption, considering κ to be constant for a small distance δt along ω. This allows us to integrate this equation as:

$$l(\mathbf{x} + \delta t\omega, \omega) = l(\mathbf{x}, \omega)(1 - \delta t\kappa(\mathbf{x})) \tag{4}$$

Let κ_{uv} be the restriction of κ to the plane, orthogonal to ω (Which means $\kappa_{uv}(\delta u, \delta v) = \kappa(\mathbf{x} + \delta u\mathbf{u} + \delta v\mathbf{v})$). We adopt the notation $p(\delta u, \delta v) = 1 - \delta t\kappa_{uv}(\delta u, \delta v)$. In the Fourier domain, Equation 4 turns into a convolution:

$$\widehat{l'} = \widehat{l} \ \otimes_{\Omega_u\Omega_v} \ \widehat{p} \tag{5}$$

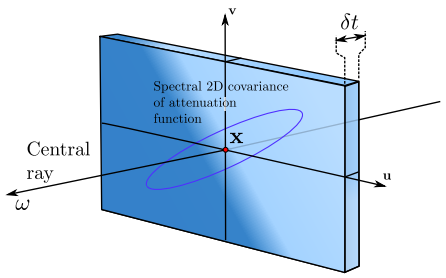


Fig. 4. Notations for the attenuation operator. We analyze the spectral covariance of the attenuation for a small travel distance δt along the central ray. Using small distances allows to assume that the attenuation is constant along ω.

In this equation, $\otimes_{\Omega_u\Omega_v}$ denotes a convolution over the spatial component only. The effect of attenuation is therefore identical to occlusion, except that the mask $p = 1 - \delta t\kappa_{uv}$ is a function taking arbitrary values in $[0, 1]$ instead of a binary function. Let A be the covariance matrix of $\widehat{p}$. Applying the covariance formula for occlusion (Figure 3), we write the covariance matrix of the convolution as the sum of the two covariance matrices:

$$\Sigma' = \Sigma + A \tag{6}$$

This equation shows that absorption transfers covariance into the spectrum of the local light field. Another way of seeing this is that the oscillations of absorption transfer into the light-field.

Computing matrix A in practice, is actually simple using Equation 2: we compute the 2D Hessian matrix $H(\mathbf{x})$ of κ in the $(\mathbf{u}, \mathbf{v})$ basis using finite differences, and diagonalize it using a 2D-rotation R. We apply the absolute value, and convert it back to the $(\mathbf{u}, \mathbf{v})$ coordinate system, using covariance rotation with the inverse rotation R^T (using Figure 3):

$$A = \frac{\delta t}{4\pi} \begin{bmatrix} \begin{bmatrix} R^T \end{bmatrix} \begin{bmatrix} |RH(\mathbf{x})R^T| \end{bmatrix} \begin{bmatrix} R \end{bmatrix} & \begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix} \\ \begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix} & \begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix} \end{bmatrix} \tag{7}$$

It follows that if a ray crosses a region with transverse sharp transitions of the attenuation function (e.g., a transverse transition from opaque to non-opaque medium, such as the one depicted on Figure 4) the attenuation matrix will represent arbitrarily large frequencies in the direction of the discontinuity; this behavior is equivalent to binary occlusion.

Note that for locally constant and linearly varying volumes, the absorption does not affect the spectral covariance of the signal. In this case the effect of attenuation is simply the change of the weight we will use when combining covariance matrices from multiple light paths that we describe in Section 5.

4.2 Scattering

In this section we derive the matrix formulation of the change in spectral covariance of a local light field, along a ray that is scattered in a participating medium. Starting from the scattering equation, we perform a first order analysis of the integral, and compute the Fourier transform of the approximated local light fields.

The scattering equation used in raytracing expresses the local increase of radiance at $\mathbf{x}$, in direction ω_s due to light scattering from all incoming directions ω according to the phase function ρ [Cerezo

et al. 2005]:

$$\frac{\partial(l(\mathbf{x}+t\omega_s,\omega_s))}{\partial t}(0) = \frac{\kappa_s}{4\pi}\int_{\omega\in S^2}\rho(\omega,\omega_s)\; l(\mathbf{x},\omega)d\omega \quad (8)$$

Integrating for a small traveling distance δt, we obtain:

$$l(\mathbf{x}+\delta t\omega_s,\omega_s) = l(\mathbf{x},\omega_s)+\frac{\delta t\kappa_s}{4\pi}\int_{\omega\in S^2}\rho(\omega,\omega_s)\; l(\mathbf{x},\omega)d\omega \quad (9)$$

When performing Monte-Carlo rendering in participating media, the sum on the right is handled by deciding with Russian Roulette whether the light path scatters or not. Consequently, to study scattering along a light path that is known to scatter, we need to deal with the integral term of the above equation only.

4.2.1 *Scattering the local light fields.* We study the implication of this equation in the 4D neighborhood of a couple of directions ω_i and ω_s, making a finite angle α (In other words, $\cos\alpha = \omega_i.\omega_s$). We derive the scattering equation for small perturbations around the incoming and outgoing directions.

We consider the incoming and outgoing light fields to be defined in correlated angular frames, for which the first angular component lies in the plane containing ω_s and ω_i, as depicted in Figure 5. Let $(\delta\theta,\delta\phi)$ and $(\delta\theta',\delta\phi')$ be the angular coordinates of the incoming and outgoing light fields in these frames, $R_{\delta\theta,\delta\phi}$ the rotation that turns the central direction into the local light field direction $(\delta\theta,\delta\phi)$. We also denote by $(\delta u,\delta v)$ (resp. $\delta u',\delta v'$) the spatial components of the 4D frame around ω_i (resp. ω_s).

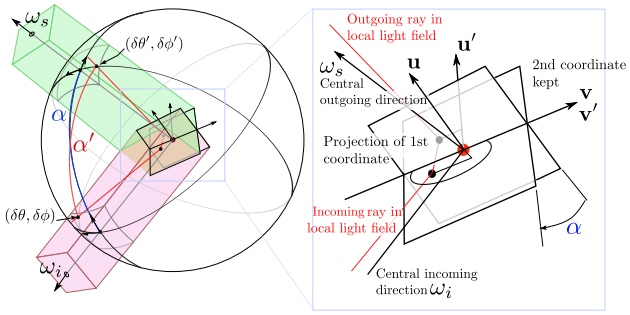


Fig. 5. Notations for scattering in 3D, with the input and output angular coordinate systems aligned. The local light field around the central incoming direction ω_i (resp. scattered direction ω_s) is parameterized with spatial and angular coordinates $\delta u,\delta v,\delta\theta,\delta\phi$ (resp. $\delta u',\delta v',\delta\theta',\delta\phi'$). The change in spatial coordinates implies a projection of the first spatial component only, scaling it by $\cos\alpha$.

With this notation in place, we express the scattered contribution of the light field around ω_i to the light field around ω_s, by restricting the integration domain in Equation 9 to local coordinates around the two vectors:

$$l_s(\delta u',\delta v',\delta\theta',\delta\phi') = \frac{\delta t\kappa_s}{4\pi}\int_{\delta\theta,\delta\phi} l_i(\delta u'\cos\alpha,\delta v',\delta\theta,\delta\phi)\rho(R_{\delta\theta,\delta\phi}\omega_i, R_{\delta\theta',\delta\phi'}\omega_s) \quad (10)$$

Note also that in the integrand l_i uses *outgoing* spatial coordinates $\delta u',\delta v'$, but *incoming* angular coordinates $\delta\theta,\delta\phi$. Reparameterizing spatial coordinates corresponds to a projection of the first spatial coordinate, resulting in the 1D scale by $\cos\alpha$ that expands the signal along the first spatial coordinate.

We suppose that $\rho(\omega_i,\omega_s)$ only depends on the relative position of the two vectors, in which case we can express it as $\rho(\cos\alpha)$. Given an input angular perturbation $(\delta\theta,\delta\phi)$ and output angular perturbation $(\delta\theta',\delta\phi')$ in the equatorial parametrization, the angle α' between those directions obeys the law of cosines (See for instance [Todhunter 1859] page 18, and Figure 5), which for small perturbations, boils down to:

$$\cos(\alpha') = \cos(\alpha+\delta\theta_i-\delta\theta_s)\cos(\delta\phi_i-\delta\phi_s)$$

We adopt the following notation for the phase function in the neighborhood of (ω_i,ω_s):

$$\overline{\rho}(\delta\theta'-\delta\theta,\delta\phi'-\delta\phi) = \rho(\cos(\alpha+\delta\theta-\delta\theta')\cos(\delta\phi-\delta\phi'))$$

Equation 10 becomes:

$$l_s(\delta u',\delta v',\delta\theta',\delta\phi') = \frac{\kappa_s\delta t}{4\pi}\int_{\delta\theta,\delta\phi} l_i(\delta u'\cos\alpha,\delta v',\delta\theta,\delta\phi)\overline{\rho}(\delta\theta'-\delta\theta,\delta\phi'-\delta\phi)$$

In the angular domain, this is a 2D convolution between the incident light field and the phase function ρ in the neighborhood of directions ω_i,ω_s.

4.2.2 *Spectrum covariance formula.* Given this formulation, going to Fourier space then follows in a straightforward manner. We simply take the Fourier transform on both sides, to get:

$$\widehat{l_s}(\Omega_u,\Omega_v,\Omega_\theta,\Omega_\phi) = \frac{\kappa_s\delta t}{4\pi\cos\alpha}\widehat{l_i}\left(\frac{\Omega_u}{\cos\alpha},\Omega_v,\Omega_\theta,\Omega_\phi\right)\widehat{\overline{\rho}}$$

To translate this relationship into covariance matrices, we apply the formulae summarized in Figure 3: the convolution adds angular bandwidth to the inverse of the covariance (and thus effectively removes angular bandwidth), and the scale by $1/\cos\alpha$ scales the covariance by $\cos^2\alpha$. The outside factor is kept away for normalization according to Equation 1:

$$\Sigma_s = ((V_\alpha\Sigma_i V_\alpha)^{-1}+S)^{-1} \quad (11)$$

where S is the covariance matrix of the scattering operator. S depends on the 2D covariance matrix σ_ρ of the windowed Fourier spectrum of $\overline{\rho}$, and V_α is the scale matrix due to the spatial reparameterization:

$$S = \begin{bmatrix} 0&0&0&0\\ 0&0&0&0\\ 0&0&\multicolumn{2}{c}{\multirow{2}{*}{σ_ρ^{-1}}}\\ 0&0&& \end{bmatrix} \quad V_\alpha = \begin{bmatrix} \cos\alpha&0&0&0\\ 0&1&0&0\\ 0&0&1&0\\ 0&0&0&1 \end{bmatrix} \quad (12)$$

To compute matrix σ_ρ, we use Equation 2, which directly gives the spectral covariance of the phase function around (ω_i,ω_s) from the Hessian matrix of $\overline{\rho}$ (we suppose that the Hessian is diagonal, without loss of generality. Otherwise, an additional rotation is needed just like what we did for absorption):

$$\sigma_\rho = \frac{1}{4\pi^2\overline{\rho}(0,0)}\begin{bmatrix} \left|\frac{\partial^2\overline{\rho}}{\partial\theta^2}(0,0)\right| & 0\\ 0 & \left|\frac{\partial^2\overline{\rho}}{\partial\phi^2}(0,0)\right| \end{bmatrix} \quad (13)$$

The effect of scattering is therefore very similar to what a BRDF does on the local light field: it removes frequencies.

It is also interesting to note that for $\alpha=\frac{\pi}{2}$, the spectrum covariance in Ω_u is totally removed by the above equation. This is because in this case, the incoming and outgoing directions are perpendicular, and therefore no variation along u on the incoming light affects the outgoing light field. Note also that for a general light

path scattering multiple times in a volume, Equation 11 needs to be interleaved with rotations to correctly align coordinate systems between two scatter events.

In summary, we proved that the effect of the scattering operator to the covariance matrix will be: a BRDF operator followed by a scaling of the spatial component. We will now give an example of how to compute S when ρ is the Henyey-Greenstein function.

4.2.3 *Covariance of the Henyey-Greenstein phase function.* There are multiple analytical models of phase functions available [Gutierrez et al. 2009]. As a practical example, we give the formulas of the spectral covariance matrix for the Henyey-Greenstein phase function, that is most common in the field. This function is defined using angle α between the incoming and outgoing directions, as

$$\rho_g(\omega_i, \omega_s) = \frac{1}{4\pi} \frac{1-g^2}{(1+g^2-2g\cos\alpha)^{\frac{3}{2}}} \quad \text{with} \quad \cos\alpha = \omega_i.\omega_s$$

We show in Appendix B that the spectral covariance of the Henyey-Greenstein function locally around ω_s is:

$$\text{cov}(\hat{\rho_g}) = \frac{1}{4\pi^2} \begin{bmatrix} |h_{11}| & 0 \\ 0 & |h_{22}| \end{bmatrix} \tag{14}$$

with

$$\begin{aligned} h_{11} &= \frac{3g(2(g^2+1)\cos\alpha + g(3\cos(2\alpha)-7)}{2(g^2-2g\cos\alpha+1)^2} \\ h_{22} &= \frac{3g\cos\alpha}{g^2-2g\cos\alpha+1} \end{aligned}$$

As expected, the covariance is isotropic for $\alpha = 0$ (*i.e.* $h_{11} = h_{22}$) since the Henyey-Greenstein function is rotationally symmetric, and is null for $g = 0$, since the function is constant in this case. We validate these equations in Figure 6 with a comparison of ground truth and predicted covariance matrices for two different values of α.

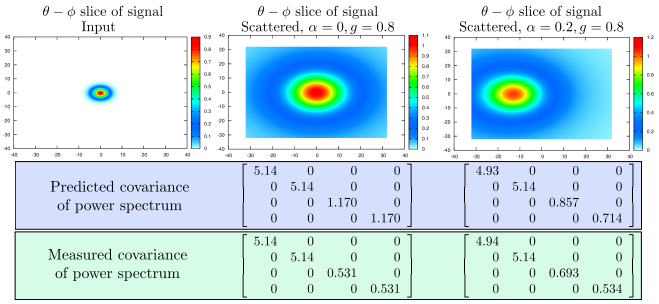


Fig. 6. Validation of Equations 11 and 14. We measure the covariance of the windowed spectrum of an input light beam (*left*) after scattering with two different angles (resp. $\alpha = 0$, *middle*, and $\alpha = 0.2$, *right*), for a Henyey-Greentein parameter $g = 0.8$. *Top row:* Angular slice of the signal (in the primal domain). *Middle row:* Predicted covariance matrix of the spectrum. *Bottom row:* measured covariance matrix, from the 4D data sampled with 64^4 in $[-1, 1]^4$. Given the order of magnitude of numbers involved in the calculation, the measured and predicted values are very close (besides, the square root of the diagonals must be compared). Our calculation is conservative and well predicts the behavior of the measured data.

Summary

In this section we have derived equations to compute the spectral covariance of the local light field along a light path inside participating media. We have shown that absorption increases frequency content and acts as the previously defined occlusion operator. Scattering low-pass filters the angular frequencies of the input local light-field with a bandwidth defined by the phase function. Thus, it acts like the BRDF operator.

5. SAMPLING AND INTEGRATION METRICS IN 3D

In Section 4, we performed an analysis of scattering and attenuation in the 4D space of local light fields along rays. In participating media, light bounces in all directions, and the covariance of a single ray cannot be used to predict the overall frequency characteristics of the light distribution. In this section we will see how to leverage the 4D local analysis to compute a set of sampling metrics in 3D, by combining the covariance from many rays.

We consider the following metrics: *image-space bandwidth* will enable efficient image space sampling and reconstruction; the *variance in the volume along a ray* will prove useful to optimize the placement of integration samples for integrating illumination along a ray; and finally, a prediction of *volumetric bandwidth* will predict the optimal size of density estimation kernels to improve volumetric integration techniques, such as progressive photon mapping,

Combining the 4D local covariance of many rays into a single 3D field also has favorable practical consequences: we favor reusing the covariance of a small subset of light paths by storing it in a buffer. However, since the spectral covariance of radiance is directional it would ideally need to be stored in a 5D buffer, a potentially computationally and memory intensive datastructure. Fortunately, a good compromise is to base our metrics on the frequency covariance of the volumetric fluence only, which requires a spatial (3D) buffer, at the cost of a very reasonable approximation.

5.1 The volumetric covariance

The volumetric covariance is the covariance of the power spectrum of the fluence in a volume. It bridges the gap between the light path formulation of covariance derived in Section 4, and our proposed practical improvements of sampling strategies in existing global illumination methods.

We define the volumetric covariance to be the 3×3 covariance matrix $\Gamma_{\mathbf{x}}$, where entry (i, j) is the ij-covariance of the Fourier transform of the fluence $F_{\mathbf{x}}$ in the neighborhood of $\mathbf{x}$:

$$(\Gamma_{\mathbf{x}})_{ij} = \frac{1}{F_{\mathbf{x}}(0)} \int \Omega_i \Omega_j \widehat{F_{\mathbf{x}}}(\mathbf{\Omega}) d\Omega \tag{15}$$

The local fluence F at an offset $\mathbf{s}$ around the point $\mathbf{x}$ is defined as:

$$F_{\mathbf{x}}(\mathbf{s}) = \int_{\omega \in S^2} l(\mathbf{x}+\mathbf{s}, \vec{\omega}) d\omega$$

In practice, the volumetric covariance is computed and stored in a voxel grid, and the size of the neighborhood considered for each voxel is the size of the voxel itself (so, $\mathbf{x} + \mathbf{s}$ is restricted to lie in the voxel).

Computation. We compute the volumetric covariance by accumulating contributions of individual light paths traversing the volume. At a point $\mathbf{x}$, the 4D spectral covariance Σ of an incident light path in direction ω carries the illumination from a very localized set of directions around ω. The 2D spatial sub-matrix of the 4D covariance of the local light field around ω is therefore a slice of the

3D covariance of the integrated radiance, in the plane orthogonal to ω.

Consequently, we compute the covariance of the fluence at $\mathbf{x}$ by summing up the 2D spatial slices of the covariance matrices of each incident light path, padded to 3D with zeroes, and rotated to match the world coordinate system. Since Σ lives in Fourier space, and the summation happens in the primal domain, submatrices need to be extracted from Σ^{-1} and inverted back to Fourier space after summation:

$$\Gamma_p = \left(\int_{\omega \in S^2} R_\omega \Sigma^{-1}|_{\delta x, \delta y} R_\omega^T I(\omega) d\omega \right)^{-1} \tag{16}$$

In this equation, the notation $\Sigma^{-1}|_{\delta x, \delta y}$ refers to the 2D spatial submatrix of matrix Σ^{-1}, while R_ω is the 3×2 matrix converting the two local spatial coordinates around ω into the three coordinates of the world. Finally, $I(\omega)$ is the normalized incident energy along incoming direction ω.

In practice, the integral in Equation 16 is computed using a classical Monte Carlo summation, as light paths in the volume cross voxels they contribute to. We do not need to explicitly compute $I(\omega)$ since it is naturally handled by the photon tracing approach: the number of path crossing a voxel is proportional to the fluence. We only record how many times each voxel was hit, for proper normalization.

5.2 Image-space covariance

We want to compute image-space covariances for adaptive sampling. The angular sub-matrix of the covariance Σ at the camera can be used to derive sampling densities and reconstruction filters for ray-tracing, at each pixel [Belcour et al. 2013].

The most straightforward method to obtain Σ for each pixel in the screen would be to accumulate covariance matrices from light paths reaching the camera, applying the theory of Section 4. While this eventually provides an unbiased estimate of the image-space covariance, it needs many light paths to obtain a reasonably noise-free estimate.

It is the spatial variations of "light intensity" in the participating medium that will show up as angular bandwidth at the camera, and after projection, as spatial bandwidth on the screen [Durand et al. 2005]. Consequently, we propose computing screen-space bandwidth from Γ. To compute Σ at the camera, we slice the volumetric

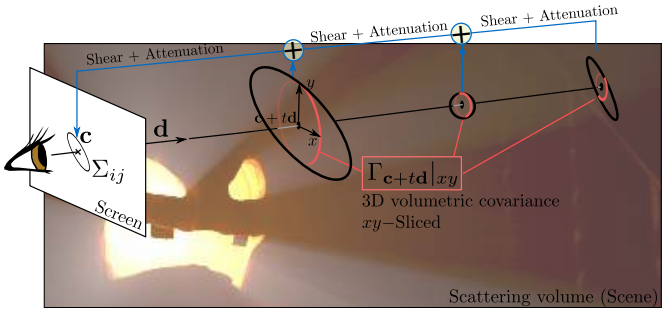


Fig. 7. To estimate the 2D spatial covariance matrix in image space, we accumulate slices of the volumetric covariance Γ along the ray using Equation 17, and the equations of attenuation 6 and 7.

covariance Γ orthogonally to camera rays, pad it to 4D with null angular covariance, and accumulate it using Equations 6 and 7 to account for the attenuation between points along the ray and the camera. At pixel p, corresponding to a ray with origin c and direction d:

$$\boxed{\Sigma(p) = \frac{1}{\int K(t)dt} \int_{t=0}^{D} K(t) T_d^T \left(A(t) + \Gamma_{\mathbf{c}+t\mathbf{d}}|_{xy}\right) T_d dt} \tag{17}$$

The normalization constant $K(t)$ accounts for how much energy is associated with each voxel in the covariance grid. The resulting matrix is a 4D covariance from which we extract the angular component at the camera. The process is illustrated in Figure 7.

Equation 17 is an approximation because it implicitly supposes that the last bounce of light before the camera has no angular covariance, meaning that the last scattering step is isotropic. In practice we found this approximation to have no visible effect.

5.3 Ray-space variance

When gathering energy in participating media, one needs to integrate illumination along rays. For each ray, we need to make sure that the spacing between integration samples avoids aliasing with respect to how much the illumination varies in the volume. That requires the variance $v(t)$ of the fluence function along the ray. The variance of the fluence in a particular direction ω is naturally given by a 1D slice of the volumetric covariance matrix, in direction ω (supposedly a unit vector):

$$\boxed{v(\mathbf{x}, \omega) = \omega^T \Gamma_{\mathbf{x}} \omega} \tag{18}$$

5.4 Optimal kernel size for density estimation

Methods based on density estimation such as photon mapping, need to carefully adapt the size of the kernel to collect photons: too small a kernel increases variance, while too large a kernel increases bias. Silverman gives an estimate of the mean integrated square error for density estimation (see Silverman's monograph [1986], page 85) which depends on the Laplacian of the estimated function. In the case of photon mapping methods the function is the fluence, and Silverman's estimate gives:

$$\text{bias}_h(\mathbf{x}) = \frac{1}{2} h^2 a \triangle F(\mathbf{x})$$

In this formula, h is the radius of the kernel used for density estimation, and a is a constant that only depends on the shape of the kernel. We use again equation 2, that links the diagonal of the covariance matrix to the absolute value of the partial second derivatives of $F(\mathbf{x})$, to obtain an upper bound on the absolute value of the Laplacian from the trace of the covariance matrix:

$$|\triangle F(\mathbf{x})| \leq 4\pi^2 F(\mathbf{x}) \text{tr}(\Gamma_{\mathbf{x}})$$

Equality holds when the second derivatives share the same sign. From this, we have an upper bound on the density estimation bias:

$$\boxed{|\text{bias}_h(\mathbf{x})| \leq 2\pi^2 h^2 \; a \; F(\mathbf{x}) \; \text{tr}(\Gamma_{\mathbf{x}})} \tag{19}$$

This equation directly gives us the largest possible kernel radius h to always keep the bias below a given threshold.

Summary. In this section we have explained how to combine the 4D spectral covariance of many rays into volumetric 3D covariance. We have derived interesting sampling metrics from volumetric covariance. In the next section, we will explain how to use these metrics to improve existing rendering algorithms in three different application scenarios.

6. IMPROVEMENT OF EXISTING SAMPLING AND RECONSTRUCTION METHODS

In this section we demonstrate the usefulness of our analysis from Section 4, and the sampling prediction metrics we derived in Section 5. We examine four different calculation steps that are involved in computational methods of global illumination in participating media, and we show that our sampling metrics can be used to accelerate them.

Our sampling metrics need the computation of volumetric 3D covariance, as defined in Section 5. To compute and store volumetric covariance, we use a voxel grid, the *covariance grid*. In the use cases below, we always read the values of Γ in that grid to compute the required metrics. All results are computed on an Intel i7-3820 with four cores at 3.60GHz per core and an NVidia GeForce GTX 680. We use 8 threads to benefit from hyperthreading. Unless noted, we use a 64^3 covariance grid. The covariance grid population algorithms run on a single thread, while we use multi-processor capabilities for volume integration (Section 6.3 and 6.4 using OpenMP) and for density estimation (Section 6.5 using CUDA).

6.1 The covariance grid

We sample light paths, and populate the covariance grid using Equation 16. We also record how many times each voxel in the grid is visited by light paths, so as to maintain information for proper normalization.

This calculation is not view-dependent. Depending on the application, we populate the covariance grid using a fixed proportion of the light paths used for the simulation (in Section 6.5), or fill it up once before the simulation (Sections 6.3 and 6.4). Figure 9 references the values used for the different scenes. For the algorithms of Section 6.3 and 6.4, ray marching and filling the covariance grid with $100,000$ light paths takes 21 seconds for a 64^3 covariance grid with the Halloween scene. We used as many as $10,000$ light paths for the Sibenik scene, as the lights are spot lights. With this amount of light paths, it took 8 seconds for ray marching and filling for the 64^3 covariance grid.

Figure 8 shows the volumetric covariance predicted by our system in three different locations of a scene showing volumetric caustics and shadows.

6.2 Efficient prediction of the Hessian of the fluence

The covariance grid naturally enables a stable computation of second derivatives of the fluence. In this section we study the benefit of estimating second derivatives from the covariance in frequency space using Equation 2, rather than trying to estimate second derivatives in the primal domain using a finite differences scheme.

We present such a comparison in Figure 10, in a simple volume containing a point light source, and for two different integration schemes: a 3-points second derivative estimate, which naturally proves to be very noisy, and a heavily filtered estimate using the second derivative of a Gaussian over 21^3 neighbor voxels. This experiment not only proves that our model gives a correct estimate of the second derivatives, but also that it converges faster than methods based on the primal domain. This is not surprising, because our method does not require explicit differentiation over the path-traced illumination the way the primal domain estimate does.

Methods that rely on linear interpolation between 3D illumination samples in the volume theoretically have an interpolation error that is proportional to the second derivatives of the illumination. This has been proven for surface irradiance caching methods [Schwarzhaupt et al. 2012], and such an error estimate outperforms existing heuristics [Jarosz et al. 2008].

Although extending our work to volumetric irradiance caching is beyond the scope of this paper, we believe that the irradiance caching error estimate based on the Hessian can be extended to volumes, with an error estimate that is proportional to the Laplacian of the fluence (see, for instance, Equations 5 and 6 in [Schwarzhaupt et al. 2012]), and where the shape of the influence regions of irradiance cache samples will be ellipsoids aligned with the eigenvectors of the Hessian of the fluence in the volume. This opens interesting research avenues toward the replacement of existing heuristics for error and influence regions for volumes [Ribardière et al. 2011], especially removing the need for special treatment of occlusion [Schwarzhaupt et al. 2012] that the covariance analysis naturally handles.

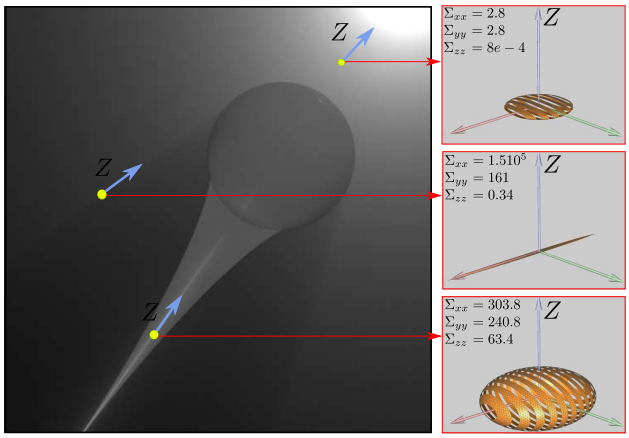


Fig. 8. The volumetric covariance Γ is the covariance of the fluence around each point in the volume. We show values of Γ from the *covariance grid*, as an ellipsoid (iso-value of the 3D Gaussian with same covariance). *Top:* in the region of diffusion, Γ is really small. *Middle:* on the shadow boundary, Γ is large along the normal to the shadow volume. *Bottom:* in the caustic, Γ is large orthogonally to the direction of the caustic (Caution: All graphs are normalized to help demonstrate the shape of the covariance. To compare the covariance quantitatively, look at the eigenvalues listed for each dimension).

		Halloween	Sibenik
Image Space	Cov. grid samples	100 000	10 000
	Step size	0.01	0.002
Eye Path	Cov. grid samples	100 000	10 000
	Step sizes	$0.1 - 0.01$	$0.1 - 0.002$
Naive	Step size	0.01	0.002
Prog. Photon Beams	α	0.7	0.7
	Samples per pass	5 000	5 000

Fig. 9. We list the different parameters used for our results section. We report the number of light paths used to fill the covariance grid, the distance between samples along the eye ray for the integration in the volume, and Progressive Photon Beams [Jarosz et al. 2011] parameters (radius reduction ratio α, and the number of light paths sampled per pass). We report only scenes that are common to the three algorithms. We used $\sigma_s = 0.1$, $\sigma_a = 0.1$ and $g = 0.5$ for the parameters of the volume. For our adaptive sampling and image space filtering algorithms, we used 8 jittered supersampling samples per pixel to obtain anti-aliasing.

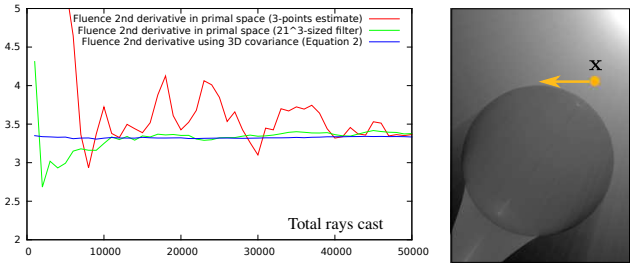


Fig. 10. Comparative estimate of the second derivative of the fluence in the primal domain (red and green curves) versus Equation 2 (blue curve), for point $\mathbf{x}$ in the direction of the arrow, as a function of the number of beams cast. In both cases we used grids of 128^3 voxels to store the covariance and the fluence. In the primal space, the second derivative is estimated in two ways from the fluence grid. *Red curve:* using a finite difference scheme between immediate neighbor voxels. *Green curve:* using a very large Gaussian filter around the measured voxel (averaging the nearby 21^3 voxels). For the *blue curve*, we simply applied Equation 2 to the covariance matrix picked at that voxel in the covariance grid, without filtering. With as low as 5000 total rays cast in the scene our estimate outperforms the costly filtered estimate in the primal domain.

6.3 Image adaptive sampling and reconstruction

An effective method for rendering images with varying local bandwidth is to compute image space converged illumination samples, and filter these samples using an appropriate 2D reconstruction kernel. For an optimal result, it is necessary to know in advance the optimal sampling density (in samples per square pixels), and the shape of the reconstruction filter at each pixel [Belcour et al. 2013] or to compute it from a subset of the light paths used in the image [Overbeck et al. 2009; Rousselle et al. 2011].

The optimal sampling densities $N(p)$ and the shape of the 2D image reconstruction filter f_p can be derived for each pixel p from the covariance $\Sigma(p)$ of the local light field at that particular pixel [Belcour et al. 2013]:

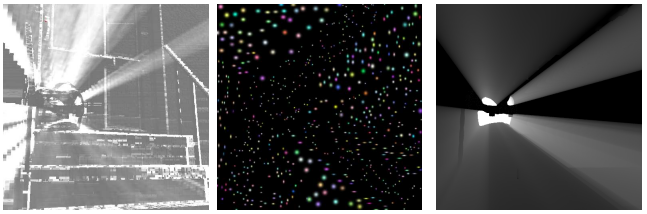
$$N(p) = k\sqrt{|\Sigma(p)|} \quad \text{and} \quad f_p(x) = e^{-\frac{1}{2}\mathbf{x}^T(\Sigma(p)^{-1})|_{x,y}\mathbf{x}} \quad (20)$$

In this expression, $\Sigma(p)^{-1}|_{x,y}$ is the spatial slice of the inverse of the spectrum covariance matrix in the image plane at pixel p. In other words, $\Sigma(p)$ is the covariance of the Gaussian whose variance matches the bandwidth of the image according to the Nyquist rate. In practice, for each pixel, we trace a single ray through the covariance grid and apply Equation 17 to compute $\Sigma(p)$.

The number of samples per square pixel is proportional to the determinant of the screen-space spectrum covariance, and the shape of the filter is obtained by slicing the covariance of the signal along the spatial dimensions at each pixel. The constant k lets us express $N(p)$ as a fraction of the total number of samples allocated for the entire image.

To compute the image, we obtain the required number of samples per pixel using Equation 20 and form an *image sampling density map*. We use this map as a probability density to draw pixels to be computed, using rejection sampling. For each pixel to compute, we estimate the radiance using path tracing. Each image sample is therefore a converged illumination value. Finally, we reconstruct the image by filtering the image samples around each pixel p with the filter f_p that is given by Equation 20.

This computation is efficient because it samples the image very sparsely when the resulting image is predicted to be smooth. Figure 11 shows the result of such a computation for the pumpkin scene. The number of computed pixels is 43% of the total number of pixels in the image. The results also show that we correctly predict the shape and size the of reconstruction filters. For more trivial scenes, the gain is even better.



Estimated density of image samples | Reconstruction filters | Reconstructed image (Single scattering only)

Fig. 11. We demonstrate the use of our prediction metrics for image space filtering and reconstruction of *single scattering*. We predict the varying density of image samples to compute using Equation 20 (*left*) as well as the shape of the Gaussian filter to reconstruct from these samples at each pixel (*middle*), to reconstruct the image (*right*).

6.4 Adaptive sampling along a ray

For each pixel that we compute, we need to integrate the illumination that is scattered towards the camera. For this we integrate the radiance after a last scattering event. Instead of using uniform samples, we adapt the sampling density to the variance of the fluence along the ray to the camera, as computed in Section 5.3. The variance is directly computed from the covariance grid using Equation 18. This allows us to place samples in regions where the path crosses rapid changes in illumination, which can be caused by volumetric shadows, for example, as illustrated in Figure 12. In practice we first uniformly sample the radiance along the eye path, and then we use additional samples in proportion to the local variance estimate along the path.

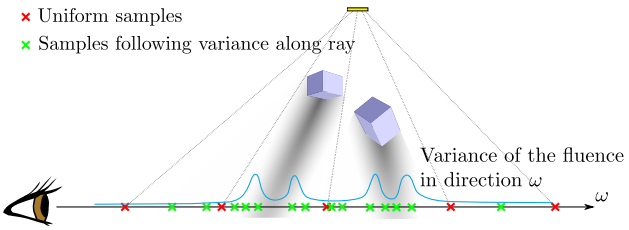


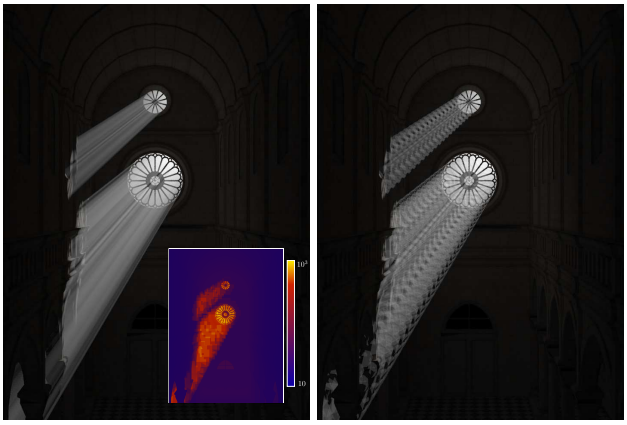
Fig. 12. Computing the amount of light reaching the camera through a given pixel requires integrating the radiance scattered by the participating medium towards the camera. Using an arbitrary number of uniformly spaced integration samples (red) might do a bad job if not accounting for the distribution of light into the volume. We propose instead to distribute samples according to the local variance of the fluence along the ray (green), in order to adaptively sample high frequency regions.

We provide a simplified algorithm for the adaptive integration used with Algorithm 1. It uses Equation 18 to evaluate the required distance between two samples in the volume.

Since our algorithm takes advantage of adaptive sampling on shadow boundaries, we are able to reduce the aliasing of light shafts caused by undersampling high frequency regions. Figure 13 shows

Algorithm 1 Our adaptive sampling algorithm compute the single scattering radiance for an eye ray defined by its position x in space and direction ω. It returns the light scattered by the volume in the interval $[0, T_{max}]$ along the ray. Our variance estimate v (Equation 18) provides a step size in the volume. Note that the last step requires special treatment as the step might be larger than the remaining integration distance. We do not show it in this example to keep a compact formulation.

function ADAPTIVEINTEGRATION($\mathbf{x}$, ω, T_{max}, $step_{min}$, $step_{max}$)
  $rad = 0$
  $t = 0$
  while $t \in [0..T_{max}]$ **do**
    $\mathbf{x}_t = \mathbf{x} + t\omega$
    $dt = \dfrac{1}{2\sqrt{v(\mathbf{x}_t, \omega)}}$
    $dt = \text{clamp}(dt, step_{min}, step_{max})$
    $rad = rad + dt\ \text{integrateRadiance}(\mathbf{x}_t, -\omega)$
    $t = t + dt$
  end while
  return rad
end function



Ours | Equal time naive

Fig. 13. We compare our variance-driven integration method to naive uniform sampling, at equal computation time (14 minutes). Our adaptive sampling clearly removes aliasing caused by the shaft from the Rose window. Inset: total number of samples used per pixel for our algorithm. (Model of the Sibenik cathedral by Marko Dabrovic).

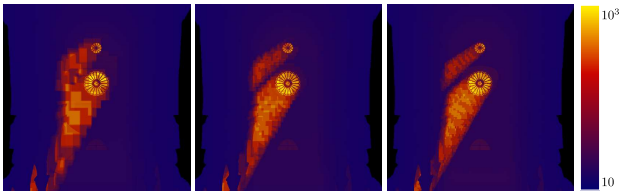
that our results on the Sibenik cathedral model outperforms uniform sampling at equal computation time, in the detailed shafts.

We summarize the timings of the image space adaptive sampling and the eye path adaptive sampling algorithm compared with an equal quality naive raytracing approach in Figure 14. Both algorithms save computation time by adapting the workload to high frequency regions.

We investigated different resolutions for the covariance grid (Figure 15). A size of 64^3 for the grid was sufficient for all our scenes. Coarser grids will provide a conservative estimation of frequencies and lead to poor performances, while finer grids will naturally increase the cost of ray marching in this structure. The cost of filling the grid is linear with the grid edge size. For the Sibenik scene using 10K light-paths, ray marching and filling took 4s for a 32^3 grid, 8s for a 64^3 grid, and 17s for a 128^3 grid. We found that a 64^3 grid provides a good trade-off between construction time, quality and time required to ray march during rendering (see Figure 15), in all our tests, except for San Miguel where we needed a 256^3 grid.

scene	Image space	Eye space	Naive
Sibenik	19m	14m	1h 40m
Halloween	7m	6m 30s	22m

Fig. 14. Our adaptive sampling and image space filtering approaches save computation time compared to a naive raytracing approach for the same quality. Eye path adaptive sampling and the naive implementation use 8 samples per pixel for antialiasing. The image space method adapts the number of samples up to this limit.



32^3 grid | 64^3 grid | 128^3 grid

Fig. 15. We analyse the impact of various resolutions of the covariance grid (32^3, 64^3 and 128^3) on the prediction of the required number of samples along camera rays. Smaller grid sizes bring more conservative results while larger grids are more costly to handle. A size of 64^3 performs well for most scenes.

6.5 Improved multiple scattering photon beams

We study the possible improvement of the convergence rate of *progressive photon beams* (PPB), to illustrate the benefits of frequency analysis.

In the original algorithm, photons are traced in the scene containing a participating medium and the paths of propagation (called beams) are stored [Jarosz et al. 2011]. Then, for each pixel, rays are shot from the camera, and the density of beams along the ray is estimated using a 2D circular kernel. This is repeated while decreasing kernel size until convergence is satisfactory.

Just like any other density estimation technique, the PPB algorithm fights between too small a reconstruction kernel—causing variance—and too large a reconstruction kernel—causing bias. Whereas the original algorithm keeps reducing the kernel sizes as more beams come along, we can afford to stop reducing it as soon as this size ensures that the density estimation bias is below a certain threshold. We know exactly when this happens from Equation 19.

During the gathering pass, for each eye ray, we test for its distance d to the beams stored (Figure 16). At the closest point to each beam along the ray, we look into the covariance matrix, and estimate the ideal gathering radius r_σ using Equation 19. We gather that beam only if:

$$d < \max(r_i, r_\sigma)$$

where, r_i is the radius given by the photon beam method for pass #i. In other words, we replace the gathering radius of progressive photon mapping by a specific radius for each pair (eye-ray, photon beam) that is adapted to the local variations of the signal. This

Table III. Images resolutions used for our rendering tests.

scene	San Miguel	Cornell Box	Sibenik	Halloween
resolution	1000×1024	512×512	714×968	512×512

adaptive radius computation stops us from decreasing the radius in regions of low bandwidth, and therefore significantly reduces variance, while keeping the error uniformly controlled. We implemented this gathering in a CUDA kernel.

We follow a pure ray tracing approach in our PPB implementation. We generate light paths and eye paths on the CPU and transfer them on the GPU. It lets us simplify the generation of rays, to follow specular paths from the camera and to avoid duplicating the scene data on the GPU. This explains the timing differences between our PPB implementation and the one of Jarosz et al. [2011].

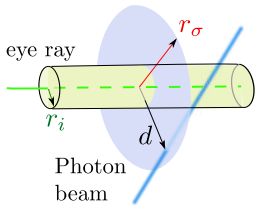


Fig. 16. Given a camera ray (green) and a beam, we use the radius r_σ, estimated by the covarianc analysis, instead of the radius r_i of the progressive photon mapping, when r_i is smaller. Using this, we gather more beams in low frequency regions and decrease the variance of the image.

We validate our improvement of progressive photon beams using the San Miguel scene (Figure 17), Halloween scene (Figure 18), Cornell box (Figure 19), and Sibenik cathedral (Figure 20) scenes. In all cases, our covariance framework correctly estimates the high frequency regions due to the illumination. San Miguel, Halloween, and Sibenik (Figures 17,18, and 20) scenes have significant indirect illumination; they prove that our method converges faster than PPB. San Miguel also demonstrate the scalability of our technique. The non-homogeneous Cornell box scene (Figure 19) validates that our analysis and filtering methods correctly handle non-constant scattering parameters. In this example, the scattering parameters are varied based on Perlin noise. Image resolutions are reported in Table III.

(a) Our improved PPB, 9.7M beams, 25min (b) Standard PPB, 10M beams, 25min

Fig. 18. The Halloween scene combines high frequency white shafts with a low frequency orange multiple scattering. Our covariance prediction allows to filter out the noise due to the diffuse component while preserving edges of shafts.

At equal computation time, we achieve a much better convergence in smoother regions of the image, while we approximately keep the same convergence in the highest frequency regions such as shaft borders and caustics, as predicted. However, our method will always eventually stop reducing the kernel size, contrary to the classical photon beam approach. It just happens later for higher frequency regions of the image.

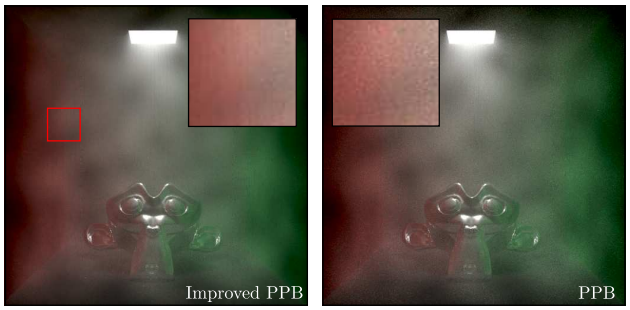


Fig. 19. In this figure, a non-homogeneous volume is illuminated. The walls of the Cornell box provide indirect color bleeding from diffuse reflections. We compare our algorithm after a run of 31 minutes (8.5M beams) with an equal time run of Progressive Photon Beams (10M beams). Our algorithm reduces noise thanks to our adaptive density estimation.

7. IMPLEMENTATION AND DISCUSSION

7.1 Implementation

Taking advantage of symmetry, 4D covariance matrices only need 10 floats to store (instead of 16) into light rays equipped with covariance information. An additional float is also needed to keep track of light intensity for proper normalization, since our definition of covariance has no units (see Equation 1). The combined covariance matrix Σ_{12} of two different paths with covariances Σ_1 and Σ_2 and intensities I_1 and I_2, after proper alignment, is:

$$\Sigma_{12} = \frac{1}{I_1 + I_2}(I_1\Sigma_1 + I_2\Sigma_2)$$

We also store, along with the covariance information, the tangent frame of the local parametrization (two 3D normalized vectors).

A photon that carries covariance is initialized at the light source and covariance is updated as the photon path is sampled [Belcour et al. 2013]. For instance, a square diffuse light source will produce rays with null angular covariance and spatial covariance that depends on the size of the square.

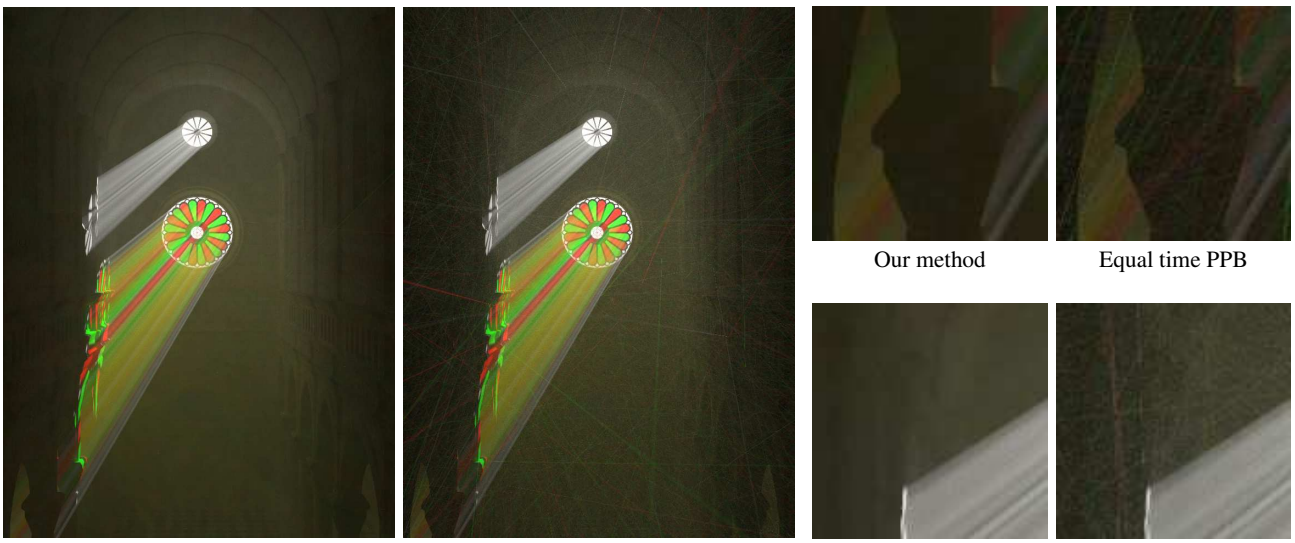
During light propagation, the covariance matrix Σ of rays is updated when the light is reflected, refracted, or occluded using the Equations of Figure 3 (see Algorithm 2). Each time a ray is scattered we transform its covariance using Equation 11, and for each absorption event, we apply Equations 6 and 7. Eventually, the various operators involved boil down to sums, products and inverses (or pseudo-inverses) of 4D covariance matrices, which is carried on very efficiently using explicit formulas.

The covariance grid uses 6-float arrays to store the individual 3D spatial covariances Γ_p and an additional float for the light intensity normalization factor per voxel. We populate the covariance grid using Equation 16, basically summing up matrices multiplied by intensity. Depending on the application, the covariance grid is either populated once, using a fixed number of rays (for e.g., adaptive reconstruction and ray-space integration) or updated during the computation (for progressive photon beams, where only 10% of the light paths carry covariance information).

We did not use ray differentials in our implementation of PPB (nor in our improvement). Jarosz et al. [2011] showed that using ray

(a) Standard PPB, 10M beams, 3h10min (b) Our improved PPB, 9.8M beams, 3h10min (c) Reference (133M beams)

Fig. 17. The full San Miguel scene with a rather indirect illumination, shows a very smooth volumetric environment, in a significantly complex geometry (2.5M polygons). In this context, our frequency prediction method makes the progressive photon beams converge much faster. Because of the size of this scene, we used 256^3-wide covariance and occlusion grids. The most severe limitation of covariance tracing in this scene was due to the occlusion grid size. Too small a grid would cause over-estimation of visibility, and conservatively populate the covariance grid with high frequencies. We used Morgan McGuire export of the San Miguel scene. As he noted on his website, `http://graphics.cs.williams.edu/data`, some geometry (chairs) and textures (walls and columns) are missing.



Our method (20M beams, 2h35min) Equal time PPB (20.5M beams) Our method Equal time PPB

Our method Equal time PPB

Fig. 20. Using our analysis metric our algorithm predicts the low frequency part of the volume where the progressive photon beam can afford to keep large collection radii, while controlling the bias. As a result, we provide a much better converged image at equal computation time than classical progressive photon beams; here, in a scene with multiple scattering.

differentials was beneficial for the convergence of specular paths from the light. But they use a constant beam radius for diffusely reflected or scattered beams. Since we compare the convergence of both algorithms for non-specular regions, this improvement of PPB was not necessary. Note that both algorithms would benefit equally from adding ray differentials.

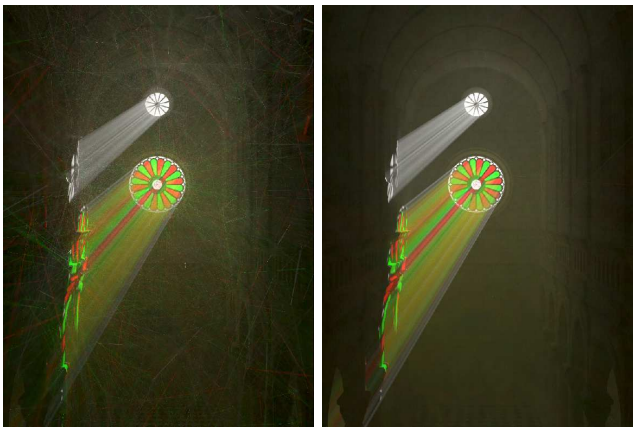
We use a sufficiently large starting radius to improve the convergence of indirect effects while still keeping convergence of the direct part. In all our examples, specular paths from the light are converged.

7.2 Limitations

The various techniques we present, based on our frequency analysis, effectively improve convergence in regions of the volume where under-sampling can be performed without loss of accuracy. If frequency is high everywhere—such as in a very highly varying medium, or in a volume where a large number of small shadow

Algorithm 2 The tracing of frequency photons is straightforward to implement. It only requires that we update the covariance matrix at specific steps. Note that it requires the ray tracing engine to compute specific information for intersection with geometry (such as local curvature). The R matrix is the factorized matrix of projection, alignment and curvature, before and after reflection. T is the covariance of the texture matrix.

function TRACEFREQUENCYPHOTON
  $\{p, \omega\} \leftarrow$ sampleLight()
  $\Sigma \leftarrow$ computeLightCovariance()
  while russianRoulette() **do**
    $p \leftarrow$ traceRay(p, ω)
    for all voxels v until hit **do**
      updateVoxelCovariance(v, Σ)
      $\Sigma \leftarrow T_d^T \Sigma T_d$
      $\Sigma \leftarrow \Sigma + O$
      $\Sigma \leftarrow \Sigma + A$
    end for
    $\omega \leftarrow$ sampleBRDF()
    $\Sigma \leftarrow R_i^T \Sigma R_i$
    $\Sigma \leftarrow \Sigma + T$
    $\Sigma \leftarrow R_o^T \left(\Sigma^{-1} + B\right)^{-1} R_o$
  end while
end function



(a) Ours, 16^3 grid, 20M beams (b) Ours, 64^3 grid, 20M beams

Fig. 21. Influence of the size of the covariance grid over the gain in convergence. A very coarse grid (16^3 in this example), will conservatively spread high frequency values into regions where the radiance is actually low frequency, failing to improve the convergence in these regions. Note: we chose to use an overly coarse grid in this example to make the effect more apparent.

rays are cast—our *a priori* analysis naturally predicts that the sampling needs to be uniformly dense. In this case, the computation of covariance information would naturally not help improve convergence.

Using volumetric covariance implies an approximation, since it neglects the angular covariance of the incident light. Our method captures variations in the volumetric fluence which, for reasonably non-specular phase functions, remains close to the variance of the radiance, while only requiring a small storage cost. In the case of a very specular phase function at the last bounce before the camera, a 3D covariance grid is likely to produce a conservative overestimation of the bandwidth. A more accurate approach would require also storing directional information into the covariance grid, and does not invalidate our frequency analysis.

The size of the covariance grid is another important limitation as it determines the scale at which we can optimize the radius reduction. A coarse grid will conservatively estimate small kernel sizes in low varying regions since high frequencies will leak outside of the region where they actually take place (This is illustrated in Figure 21).

The paraxial approximation used by the frequency analysis limits the capacity of our predictions to describe the full directional variations of light with a few photons. The paraxial approximation is valid for angles below one degree. However, using our estimates, based on this assumption, to estimate light variations works in practice.

7.3 Discussion

Performing our analysis in the Fourier domain around light paths brings us a local characterization of the signal's bandwidth. Wavelets also bring bandwidth estimation of a signal. However, they perform a local analysis at the cost of a making operations like convolution and scaling much more complex. In our case, localization is already brought by windowing our analysis around a particular point of the signal, and the Fourier transform appears to be the most simple approach to characterize bandwidth. Polynomial bases in turn, don't offer simple expressions for convolution and scale.

Our theoretical derivations perform first order approximations of the scattering and absorption operators, as opposed to first order approximations of the spectra. Linearly approximating the spectra would be meaningless. In our framework, spectra contain all possible frequencies. The only assumption made is the paraxial approximation [Durand et al. 2005].

As a comparison to the work of Yue et al. [2010], our adaptive sampling strategy is based on the variance of the illumination, whereas Yue's algorithm is based on the maximum variance of the density in the medium along a light path. Therefore we avoid wasting time oversampling highly varying regions with low energy. While adaptive sampling techniques are usually based on an *a posteriori* estimation of the energy (sometimes the bandwidth) of the signal, we base our sampling on an *a priori* prediction of the variance of the signal.

Kulla et al. [2012] propose strategies for importance sampling participating media. Our approach is complementary since we believe that importance sampling metrics can be derived from the volumetric covariance. Combining our covariance prediction tool with their importance sampling metrics would allow to drive the importance sampling by the actual distribution of energy in the solution.

Keeping a large radius for progressive photon beams could slow down the selection process, if using an acceleration structure (such as a KD-tree [Sun et al. 2010]), since this increases the branching factor of the KD-tree search. In our CUDA implementation, which follows the method described by Jarosz et al., not having an acceleration structure removes this penalty.

When filling the covariance grid, we do not need to record a directional distribution of incident illumination to weight the covariance contributions from incident rays, since those rays are path-traced and arrive with a probability that is already proportional to the illumination. Consequently, even in its current and simple implementation, the covariance grid allows us to perform anisotropic filtering of light beams. This is visible in the caustic scene (Fig-

ure 8) where we can see that the covariance estimates capture the directional discontinuities of the light distribution.

We chose to apply our framework to accelerate progressive photon beams rather than the more recent progressive virtual beam lights [Novák et al. 2012]. The two methods, however, use a similar iterative radius reduction scheme. Therefore, one can expect to improve on the latter the same way we improve on progressive photon beams, using our radius reduction stopping criterion.

Similar to Novak et al. [2012], we could use our method to only reconstruct the multiple scattering component of the illumination and not store the specular contribution into the covariance grid. Although did not do that, we still produce convincing results as our method adapts to any effect (be it direct of indirect). Treating indirect scattering independently would enhance the speedup factor of our method as the reconstructed signal would be of even lower frequency. But this would increase the required engineering for the implementation (multiple photon maps would be required).

Our improved progressive photon beams method removes the need to finely tune the parameters of the radius reduction. This, in a way, is similar to the Adaptive Progressive Photon Mapping of Kaplanyan and Daschbacher [2013]. One notable difference is that our method does not need to evaluate the Hessian of the radiance using density estimation, and as shown in Section 6.2 our estimate is much more robust.

Adding the time analysis [Belcour et al. 2013] is straightforward but currently limited to rigid motion. The analysis of time varying media is also possible, but beyond the scope of this paper. Time analysis could be implemented using the 3D covariance grid by integrating the time dimension. This way, motion events are blurred according to the resulting appearance. A 4D grid would be necessary to perform temporal coherent filtering. Adding depth of field is also orthogonal to this work, but we expect it would not cause particular issues.

8. CONCLUSION

We proposed the first extension to participating media, of the Fourier analysis of local light fields. This is a very important problem that is amenable to performance acceleration, since participating media typically has lower frequencies.

We show how to extend the use of covariance matrices in a principled manner to represent the spectrum of the light field including scattering and absorption. We derive the equations to combine the information carried by each light path into a set of 3D frequency prediction metrics, and to compute them from a common quantity: the volumetric covariance, stored in a grid. We used these metrics to improve the convergence and efficiency of image-space adaptive sampling and reconstruction, camera ray integration, and for accelerating the progressive photon beam approach.

Several future avenues of research exist. While we demonstrated the use of this analysis for the gather part of photon mapping, our frequency estimate could also be used to drive photon tracing, like, for example, using a function of frequency as the acceptance function of Metropolis photon sampling [Fan et al. 2005]. Another interesting research avenue would be to extend our analysis to oriented media [Jakob et al. 2010].

We did not use an adaptive covariance grid. To do that, the local resolution of the covariance grid needs to be adapted to the variations of the covariance information to be stored, and would spare the need to specify an initial resolution. We leave this question to future work.

ACKNOWLEDGMENTS

The authors thank user Mundomupa from blendswap.com for the design of the pumpkin model (used in the Halloween scene) and Jeremy Birn for the design of the house front (used in the Halloween scene).

The authors also thank Romain Pacanowski for this helpful comments during the redaction of this work.

This work was supported in part by a grant ANR-11-BS02-006 "ALTA".

REFERENCES

BAGHER, M., SOLER, C., SUBR, K., BELCOUR, L., AND HOLZSCHUCH, N. 2012. Interactive rendering of acquired materials on dynamic geometry using bandwidth prediction. In *Proceedings of the 2012 ACM Siggraph Symposium on Interactive 3D Graphics and Games - I3D*. 127–134.

BARAN, I., CHEN, J., RAGAN-KELLEY, J., DURAND, F., AND LEHTINEN, J. 2010. A hierarchical volumetric shadow algorithm for single scattering. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2010) 29,* 6 (Dec.), 178:1–178:10.

BELCOUR, L. AND SOLER, C. 2011. Frequency based kernel estimation for progressive photon mapping. In *SIGGRAPH Asia 2011 Posters*. ACM, 1.

BELCOUR, L., SOLER, C., SUBR, K., HOLZSCHUCH, N., AND DURAND, F. 2013. 5D Covariance Tracing for Efficient Defocus and Motion Blur. *ACM Transactions on Graphics 32,* 3 (July), 31:1–31:18.

CEREZO, E., PEREZ-CAZORLA, F., PUEYO, X., SERON, F., AND SILLION, F. X. 2005. A survey on participating media rendering techniques. *The Visual Computer 21,* 5 (June), 303–328.

CHEN, J., BARAN, I., DURAND, F., AND JAROSZ, W. 2011. Real-time volumetric shadows using 1D min-max mipmaps. In *Symposium on Interactive 3D Graphics and Games on - I3D 2011*. ACM, 39–46.

DAVE, J. V. 1970. Coefficients of the Legendre and Fourier series for the scattering functions of spherical particles. *Applied Optics 9,* 8, 1888–1896.

DAVE, J. V. AND GAZDAG, J. 1970. A modified Fourier transform method for multiple scattering calculations in a plane parallel Mie atmosphere. *Applied Optics 9,* 6, 1457–1466.

DUDERSTADT, J. AND MARTIN, W. 1979. *Transport Theory*. Wiley-Interscience Publications. Wiley.

DURAND, F., HOLZSCHUCH, N., SOLER, C., CHAN, E., AND SILLION, F. X. 2005. A frequency analysis of light transport. *ACM Transactions Graphics (Proceedings of ACM SIGGRAPH 2005) 24,* 3 (July), 1115–1126.

EGAN, K., DURAND, F., AND RAMAMOORTHI, R. 2011. Practical filtering for efficient ray-traced directional occlusion. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2011) 30,* 6 (Dec.), 180:1–180:10.

EGAN, K., HECHT, F., DURAND, F., AND RAMAMOORTHI, R. 2011. Frequency analysis and sheared filtering for shadow light fields of complex occluders. *ACM Transactions on Graphics 30,* 2 (Apr.), 1–13.

EGAN, K., TSENG, Y.-T., HOLZSCHUCH, N., DURAND, F., AND RAMAMOORTHI, R. 2009. Frequency analysis and sheared reconstruction for rendering motion blur. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2009) 28,* 3, 93:1–93:13.

ENGELHARDT, T. AND DACHSBACHER, C. 2010. Epipolar sampling for shadows and crepuscular rays in participating media with single scattering. In *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games - I3D*. 119–125.

ENGELHARDT, T., NOVAK, J., AND DACHSBACHER, C. 2010. Instant Multiple Scattering for Interactive Rendering of Heterogeneous Participating Media. Tech. rep., Karlsruhe Institut of Technology.

FAN, S., CHENNEY, S., AND CHI LAI, Y. 2005. Metropolis photon sampling with optional user guidance. In *Proceedings of the 16th Eurographics Symposium on Rendering Techniques 2005*. Eurographics Association, 127–138.

GIRAUD, B. AND PESCHANSKI, R. 2006. On positive functions with positive Fourier transforms. *Acta Physica Polonica B 37,* 2 (February), 331–343.

GORTLER, S. J., GRZESZCZUK, R., SZELISKI, R., AND COHEN, M. 1986. The Lumigraph. In *Proceedings of ACM SIGGRAPH 1996*. ACM, 43–54.

GUTIERREZ, D., JENSEN, H. W., JAROSZ, W., AND DONNER, C. 2009. Scattering. In *ACM SIGGRAPH Asia 2009 Courses*. 15:1–15:620.

IGEHY, H. 1999. Tracing ray differentials. In *Proceedings of ACM SIGGRAPH 1999*. ACM, 179–186.

IHRKE, I., ZIEGLER, G., TEVS, A., THEOBALT, C., MAGNOR, M., AND SEIDEL, H.-P. 2007. Eikonal rendering: efficient light transport in refractive objects. *26,* 3 (July).

ISHIMARU, A. 1997. *Wave Propagation and Scattering in Random Media*. IEEE Press and Oxford University Press.

JAKOB, W., ARBREE, A., MOON, J. T., BALA, K., AND MARSCHNER, S. 2010. A radiative transfer framework for rendering materials with anisotropic structure. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2010) 29,* 4 (July), 53:1–53:13.

JAROSZ, W., DONNER, C., ZWICKER, M., AND JENSEN, H. W. 2008. Radiance caching for participating media. *ACM Transactions on Graphics 27,* 1 (Mar.), 7:1–7:11.

JAROSZ, W., NOWROUZEZAHRAI, D., SADEGHI, I., AND JENSEN, H. W. 2011. A comprehensive theory of volumetric radiance estimation using photon points and beams. *ACM Transactions on Graphics 30,* 1 (Feb.), 5:1–5:19.

JAROSZ, W., NOWROUZEZAHRAI, D., THOMAS, R., SLOAN, P.-P., AND ZWICKER, M. 2011. Progressive photon beams. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2011) 30,* 6 (Dec.), 181:1–181:12.

JAROSZ, W., ZWICKER, M., AND JENSEN, H. W. 2008. The beam radiance estimate for volumetric photon mapping. *Computer Graphics Forum 27,* 2 (Apr.), 557–566.

JENSEN, H. W. AND CHRISTENSEN, P. H. 1998. Efficient simulation of light transport in scences with participating media using photon maps. In *Proceedings of ACM SIGGRAPH 1998*. 311–320.

KAJIYA, J. T. AND VON HERZEN, B. P. 1984. Ray tracing volume densities. *ACM Computer Graphics (Proceedings of ACM SIGGRAPH 1984) 18,* 3 (July), 165–174.

KAPLANYAN, A. S. AND DACHSBACHER, C. 2013. Adaptive progressive photon mapping. *ACM Transactions on Graphics 32,* 2 (Apr.), 16:1–16:13.

KIM, A. D. AND MOSCOSO, M. 2003. Radiative transfer computations for optical beams. *Journal Computational Physics 185,* 1 (Feb.), 50–60.

KNAUS, C. AND ZWICKER, M. 2011. Progressive photon mapping: A probabilistic approach. *ACM Transactions on Graphics 30,* 3 (May), 25:1–25:13.

KULLA, C. AND FAJARDO, M. 2012. Importance sampling techniques for path tracing in participating media. *Computer Graphics Forum (Proceedings of EGSR 2012) 31,* 4 (June), 1519–1528.

LAFORTUNE, E. P. AND WILLEMS, Y. D. 1993. Bi-directional path tracing. In *Proceedings of third international conference on computational graphics and visualization techniques*. 145–153.

LAFORTUNE, E. P. AND WILLEMS, Y. D. 1996. Rendering participating media with bidirectional path tracing. In *Proceedings of the Eurographics Workshop on Rendering Techniques*. Springer-Verlag, 91–100.

LEHTINEN, J., AILA, T., CHEN, J., LAINE, S., AND DURAND, F. 2011. Temporal light field reconstruction for rendering distribution effects. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2011) 30,* 4 (July), 55:1–55:12.

LEVOY, M. AND HANRAHAN, P. 1986. Light field rendering. In *Proceedings of ACM SIGGRAPH 1996*. ACM, 31–42.

MEHTA, S., WANG, B., AND RAMAMOORTHI, R. 2012. Axis-aligned filtering for interactive sampled soft shadows (proceedings of acm siggraph asia 2012). *ACM Transactions on Graphics 31,* 6 (Nov), 163:1–163:10.

NOVÁK, J., NOWROUZEZAHRAI, D., DACHSBACHER, C., AND JAROSZ, W. 2012. Progressive virtual beam lights. *Computer Graphics Forum (Proceedings of EGSR 2012) 31,* 4 (June), 1407–1413.

NOVÁK, J., NOWROUZEZAHRAI, D., DACHSBACHER, C., AND JAROSZ, W. 2012. Virtual ray lights for rendering scenes with participating media (proceedings of acm siggraph 2012). *ACM Transactions on Graphics 31,* 4 (July), 60:1–60:11.

OVERBECK, R. S., DONNER, C., AND RAMAMOORTHI, R. 2009. Adaptive wavelet rendering. In *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2009)*. Vol. 28. 140:1–140:12.

PAULY, M., KOLLIG, T., AND KELLER, A. 2000. Metropolis Light Transport for Participating Media. *Proceedings of the Eurographics Workshop on Rendering Techniques*.

RAMAMOORTHI, R., ANDERSON, J., MEYER, M., AND NOWROUZEZAHRAI, D. 2012. A theory of monte carlo visibility sampling. *ACM Transactions on Graphics 31,* 5 (Sept.), 121:1–121:16.

RIBARDIÈRE, M., CARRÉ, S., AND BOUATOUCH, K. 2011. Adaptive records for volume irradiance caching. *The Visual Computer 27,* 6-8 (Apr.), 655–664.

RITCHIE, B., DYKEMA, P. G., AND BRADDY, D. 1997. Use of fast-Fourier-transform computational methods in radiation transport. *Physical Reiew E 56,* 2 (Aug), 2217–2227.

ROUSSELLE, F., KNAUS, C., AND ZWICKER, M. 2011. Adaptive sampling and reconstruction using greedy error minimization. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2011) 30,* 6 (Dec.), 159:1–159:12.

RYBICKI, G. B. 1971. The searchlight problem with isotropic scattering. *Journal of Quantitative Spectroscopy and Radiative Transfer 11,* 6.

SCHWARZHAUPT, J., JENSEN, H. W., AND JAROSZ, W. 2012. Practical hessian-based error control for irradiance caching. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2012) 31,* 6 (Nov.), 193:1–193:10.

SILVERMAN, B. 1986. *Density Estimation for Statistics and Data Analysis*, 1 ed. Chapman and Hall/CRC.

SOLER, C., SUBR, K., DURAND, F., HOLZSCHUCH, N., AND SILLION, F. X. 2009. Fourier depth of field. *ACM Transactions on Graphics 28,* 2, 18:1–18:12.

SUN, X., ZHOU, K., LIN, S., AND GUO, B. 2010. Line space gathering for single scattering in large scenes. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2010) 29,* 4 (July), 54:1–54:8.

TODHUNTER, I. 1859. *Spherical Trigonometry, for the use of Colleges and Schools*. Macmillan and Company.

VEACH, E. AND GUIBAS, L. J. 1997. Metropolis light transport. In *Proceedings of ACM SIGGRAPH 1997*. ACM, 65–76.

WALTER, B., ARBREE, A., BALA, K., AND GREENBERG, D. P. 2006. Multidimensional lightcuts. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2006) 25,* 3 (July), 1081–1088.

WALTER, B., FERNANDEZ, S., ARBREE, A., BALA, K., DONIKIAN, M., AND GREENBERG, D. P. 2005. Lightcuts: a scalable approach to illumination. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2005) 24,* 3 (July), 1098–1107.

WALTER, B., KHUNGURN, P., AND BALA, K. 2012. Bidirectional Lightcuts. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2012) 31,* 4 (July), 59:1–59:11.

WALTER, B., ZHAO, S., HOLZSCHUCH, N., AND BALA, K. 2009. Single scattering in refractive media with triangle mesh boundaries. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2009) 28,* 3 (July), 92:1–92:8.

WAND, M. AND STRASSER, W. 2003. Multi-resolution point-sample raytracing. In *Proceedings of Graphics Interface 2003*.

YUE, Y., IWASAKI, K., CHEN, B.-Y., DOBASHI, Y., AND NISHITA, T. 2010. Unbiased, adaptive stochastic sampling for rendering inhomogeneous participating media. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2010) 29,* 6 (Dec.), 177:1–177:8.

APPENDIX

A. LAPLACIAN FROM COVARIANCE OF SPECTRUM

Let f be a function defined over a 4D domain. Following basic Fourier theory, the DC of the second derivative is the integral of its spectrum:

$$\forall i,j \;\; \frac{\partial^2 f}{\partial x_i \partial x_j}(0) \;=\; \int \widehat{\frac{\partial^2 f}{\partial x_i \partial x_j}}(\mathbf{\Omega}) d\mathbf{\Omega}$$

...where the integral is carried over the entire 4D Fourier domain. We expand the Fourier transform of the derivative over each variable:

$$\begin{aligned}\forall i,j \;\; \frac{\partial^2 f}{\partial x_i \partial x_j}(0) \;&=\; \int (2\pi i \Omega_i)(2\pi i \Omega_j)\widehat{f}(\mathbf{\Omega}) d\mathbf{\Omega} \\ &=\; -4\pi^2 \int \Omega_i \Omega_j \widehat{f}(\mathbf{\Omega}) d\mathbf{\Omega}\end{aligned}$$

This formulation almost fits the covariance of the power spectrum. It actually does when $\hat{f}(\Omega)$ is real and positive, and we miss the normalization factor (which is $\int \hat{f} = f(0) = 1$). There exist a wide class of such functions [Giraud and Peschanski 2006], the simplest of which are Gaussians. In this case we have:

$$-4\pi^2 f(0) \Sigma_{ij}(\mathbf{x}_0) = \frac{\partial^2 f}{\partial x_i \partial x_j}$$

In practice, that means that we can approximate the covariance of the power spectrum of a function in a small window around a point, as soon as the function is close to the osculating Gaussian at that point. For 4D functions, the covariance matrix of the windowed power spectrum of the function around a particular point (corresponding to $\mathbf{x} \rightarrow f(\mathbf{x}_0 + \mathbf{x})$) is therefore well approximated by the following diagonal matrix in the frame of the principal curvatures:

$$\Sigma(\mathbf{x}_0) \approx \frac{1}{4\pi^2 |f(0)|} \left\| \;\; \left| \frac{\partial f^2}{\partial x_i^2}(\mathbf{x}_0) \right| \;\; \right\|_{ii}$$

In the most general case, the Hessian matrix must be diagonalised before taking the absolute values on the diagonal. Similarly, we have:

$$\sum_i \left| \frac{\partial^2 f}{\partial x_i^2} \right| \approx 4\pi^2 f(\mathbf{x}_0) \mathrm{Tr}(\Sigma(\mathbf{x}_0)) \tag{21}$$

B. LOCAL FOURIER COVARIANCE OF H-G FUNCTION

The Henyey-Greenstein phase function is defined by

$$\rho_g(\omega_i, \omega_s) = \frac{1}{4\pi} \frac{1-g^2}{(1+g^2-2gc)^{\frac{3}{2}}} \;\text{ with }\; c = \omega_i.\omega_s = \cos\alpha$$

We are interested in the 2D covariance of the windowed spectrum of function $\overline{\rho}$, defined by

$$\overline{\rho}(\delta\theta, \delta\phi) = \rho_g(\cos(\alpha + \delta\theta)\cos(\delta\phi))$$

We recall that the 2D covariance matrix of a phase function ρ, noted σ_ρ, is defined by Equation 13:

$$\sigma_\rho = \frac{1}{4\pi^2 \overline{\rho}(0,0)} \begin{bmatrix} \left| \frac{\partial^2 \overline{\rho}}{\partial \theta^2}(0,0) \right| & 0 \\ 0 & \left| \frac{\partial^2 \overline{\rho}}{\partial \phi^2}(0,0) \right| \end{bmatrix}$$

Taking the second derivatives of $\overline{\rho}$ and evaluating them at $\delta\theta = \delta\phi = 0$ gives:

$$\begin{aligned}\frac{\partial^2 \overline{\rho}}{\partial \delta\theta^2}(0,0) \;&=\; \frac{3g(g^2-1)(2(g^2+1)\cos\alpha + g(3\cos(2\alpha)-7))}{8\pi(g^2-2g\cos\alpha+1)^{7/2}} \\ \frac{\partial^2 \overline{\rho}}{\partial \delta\theta \partial \delta\phi}(0,0) \;&=\; 0 \\ \frac{\partial^2 \overline{\rho}}{\partial \delta\phi^2}(0,0) \;&=\; \frac{3g(g^2-1)\cos\alpha}{4\pi(g^2-2g\cos\alpha+1)^{5/2}}\end{aligned}$$

The Hessian being diagonal, we don't need additional rotations before taking the absolute value. When $\alpha = 0$, the second derivatives match (to $3g(g+1)/(4\pi(g-1)^4)$), as it is expected for a rotationally symmetric function. To get the covariance in equation 14, we further divide by $4\pi^2 \overline{\rho}(0,0) = 4\pi^2 \rho_g(\alpha)$.