



New Representations toward more User-Friendly 3D Synthesis

Xavier Granier

► To cite this version:

Xavier Granier. New Representations toward more User-Friendly 3D Synthesis. Human-Computer Interaction [cs.HC]. Université Sciences et Technologies - Bordeaux I, 2009. tel-00434045

HAL Id: tel-00434045

<https://theses.hal.science/tel-00434045>

Submitted on 20 Nov 2009

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Habilitation à Diriger des Recherches

Présentée à

L'UNIVERSITÉ BORDEAUX 1

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET D'INFORMATIQUE

Par **Xavier Granier**

New Representations toward more User-Friendly 3D Synthesis

Soutenue le : 19 octobre 2009

Après avis des rapporteurs :

Frédo Durand	Associate professor (MIT)
Mathias Paulin	Professeur (Université Paul Sabatier)
Roberto Scopigno	Research director (CNR Pisa)

Devant la commission d'examen composée de :

Maylis Delest	Professeur (Université de Bordeaux)	Présidente
Frédo Durand	Associate professor (MIT)	Rapporteur
Mathias Paulin	Professeur (Université Paul Sabatier)	Rapporteur
Roberto Scopigno	Research director (CNR Pisa)	Rapporteur
Christophe Schlick	Professeur (Université de Bordeaux)	Examineur
Pascal Guitton	Professeur (Université de Bordeaux)	Examineur

À Mathieu Perrot
Qui m'offre la joie d'être son parrain.

À Blanche et Ivan Granier
Qui m'offrent leurs regards d'enfants.

À Jean Granier
Qui nous conduit vers la curiosité scientifique et littéraire.

À tous ceux, parents, famille et amis
Qui font preuve de patience face à mes longues digressions historico-politiques.

Acknowledgments

All the results presented in this document would not have been achieved without the direct or indirect participation of a large number of persons. I would like to declare my gratitude to all of them.

Tous les résultats présentés dans ce document n'auraient pas été possible sans la participation directe ou indirecte d'un grand nombre de personnes. Je voudrais déclarer à chacun d'eux toute ma gratitude.

À l'origine de ces travaux, il y a George Drettakis et Claude Puech qui m'ont offert cette extraordinaire chance de faire une thèse, qui de plus est, avec eux. À ces remerciements, je voudrais rajouter toute l'équipe iMAGIS qui constituait ce creuset humain et fertile pour chacun de nous. Maintenant que je suis de l'autre côté de la barre, que je découvre les bonheurs et les difficultés de la tâche d'encadrant, j'adresse encore plus particulièrement un grand MERCI à George, pour avoir eue confiance en moi et m'avoir fait participer à la création du projet REVES.

I would like also particularly to thank peoples from the IMAGER laboratory in Vancouver. Their friendship and rare humanity have created the conditions of an amazing experience, both personal and scientific. I own them all these two wonderful years in Canada. My particular gratitude goes to Wolfgang Heidrich who offers me this position: he is very representative of the whole group, since he is a rare combination of kindness and scientific talent. I am also grateful to INRIA for continuing to support our collaboration with the LIGHT associated team.

Me voilà arrivé aux temps présents, dans l'équipe IPARLA. Encore une fois, rien n'aurait pu se faire sans la confiance qui m'a été accordée par de nombreuses fois. Je voudrais particulièrement remercier Pascal Guitton pour son support et pour l'exemple qu'il me procure dans la motivation du groupe. Je voudrais rajouter à ces remerciements Christophe Schlick, pour l'assise scientifique de nos discussions qui me pousse à aller toujours plus loin. Je voudrais remercier chacun des membres pour l'amitié que vous m'avez offerte: elle est pour moi des plus essentielles. Un remerciement plus prononcé pour Gerald Point et Gwenola Thomas qui m'ont accueilli dans leur bureau à mon arrivée.

Il est bien sûr inenvisageable que je puisse oublier dans ces remerciements les étudiants qui ont remis leur thèse entre mes mains. Sans eux, je ne serais rien ! Je voudrais ainsi remercier Florian Levet, Romain Pacanowski, Julien Hadim et Romain Vergne. J'espère avoir joué mon rôle de "coach" d'une manière convenable.

I wish to particularly acknowledge the collaborators whose works have lead to parts of this document content. I own their confidence and friendship to Cyrille Damez and Zhang Hongxin.

Je voudrais aussi particulièrement remercier les collaborateurs qui ont permis la création d'une partie de ce document. Je suis donc redevable de leur confiance et amitié à Cyrille Damez et Zhang Hongxin.

Il me serait impardonnable de ne pas remercier ceux qui m'entourent depuis si longtemps, ceux sans qui tout cela ne serait pas possible. Je veux bien sûr parler de mes parents pour l'exemple qu'ils procurent dans le respect de chacun et la curiosité aux différentes cultures, et pour leur amour inconditionnel. Je parle avec tout autant de force des mes frères, Luc et Fabien, qui par leurs richesses humaines constituent pour moi un socle irremplaçable. Et enfin, je voudrais rajouter ceux que je considère aussi comme mes frères, Benjamin et Lionel.

Abstract

*In this document, we explore the different steps usually required for image synthesis. The main focus of this exploration is to take as much as possible users into account. The creation of 3D images relies first on the definition of a collection of 3D models and on their relative positions. For this purpose, we explore **sketch-based modeling tools** for 3D meshes, extending the use of shape curves from silhouettes to profile, and exposing all the different steps of the reconstruction process together with the underlying representation. We also explore the use of such tools when users are standing in front of large displays.*

*Once the object geometry is defined, users have to work on designing its appearance, result of the interaction of light with its reflective material and geometry. We introduce a new **BRDF model and its modeling tools** to extend the range of possible lighting effects. Since the object shape participate also the shading perception, we also introduce a **shape descriptor** to extract continuous information of convexity and curvedness. This real-time analysis is used to re-introduce them in different shading styles.*

*Finally, we introduce a new lighting representation that may be used for user edition of global illumination. This representation is based on a **regular grid of vector quantities** storing the incoming lighting. The combination of volumetric and vector representation make it suitable for all algorithms that have to deal with the geometric complexity of 3D scenes.*

keywords

Image Synthesis; Geometry Modeling; Sketching and User-Editon; BRDF; Global Illumination; Expressive and Non-Photorealistic Rendering;

Dans ce document, nous explorons les différentes étapes habituellement requises pour la synthèse d'image. Le principal but de cette exploration est de prendre en compte autant que possible l'utilisateur. La création d'images 3D nécessite en premier lieu la définition d'une collection de modèles 3D et de leur position relative. Pour cet objectif, nous explorons les techniques de **modélisation 3D par esquisses** pour des maillages 3D, en étendant l'utilisation de courbe de la définition de la silhouette à celle du profil de la forme, et en exposant toutes les différentes étapes du processus de reconstruction ainsi que la représentation interne. Nous explorons aussi l'utilisation de tels outils lorsque l'utilisateur se trouve face à de grands écrans.

Une fois la géométrie d'un objet définie, un utilisateur doit définir son apparence résultant de l'interaction de la lumière avec ses propriétés de réflexion et sa géométrie. Nous introduisons un nouveau **modèle de BRBF et ses outils de modélisation** pour étendre les effets lumineux possibles. Puisque la forme de l'objet influence la perception de son éclairage, nous introduisons aussi un **descripteur de forme** pour extraire des informations continues de convexité et de courbure. Cette analyse en temps-réel est utilisée pour re-introduire ces informations dans différents styles de rendu.

Finalement, nous introduisons une nouvelle représentation pour l'éclairage qui offre un grand potentiel en terme d'édition pour l'éclairage global. Cette représentation est basée sur une **grille régulière de vecteurs d'irradiance** qui encode l'éclairage incident. La combinaison d'une approche volumique et vectorielle rend cette méthode bien adaptée pour tous les algorithmes qui doivent faire face à la complexité géométrique des scènes 3D.

keywords

Synthèse d'images; Modélisation géométrique; Modélisation par esquisse and edition; BRDF; Éclairage global; Rendu expressif et non-photorealiste;

Contents

Introduction	1
1 Sketching 3D Models	5
1.1 Context and Previous Work	5
1.2 Profile Curves for Enhanced Free-Form Modeling	8
1.3 3D Positioning for Large displays	16
1.4 Conclusion	22
2 Local Illumination and Shading	25
2.1 User-Designed Glossy and Specular BRDF	26
2.2 Shape Depiction through Expressive Shading	36
2.3 Conclusion	48
3 Global Illumination	49
3.1 Previous Work	49
3.2 Irradiance Vector Grid	58
3.3 Applications and Results	62
3.4 Conclusion	69
4 Conclusion & Future Research	71
4.1 Contributions	71
4.2 Future Research: General Approach	72
4.3 Specific Context: Cultural Heritage	75
A Global Control on Colored Reflection	77
A.1 Motivations	77
A.2 Previous Work	79
A.3 Bilinear Reflection Model	80
A.4 New Reflection Operators	81
A.5 Coefficients Estimation: $k_{d,e}^c$	82
A.6 Reflection Behavior Control	85
A.7 Applications and Results	86
A.8 Conclusion	89

B	Vectorial Definitions	91
B.1	Definitions	91
B.2	Bases for Directional Vectors	93
B.3	Extending Radiosity Equation	95
B.4	Irradiance Vector for Environment-maps	97
C	Curriculum Vitae	99
	Bibliography	109

Introduction

Computer Graphics context

In we look back into the history of Computer Graphics, we may notice an interesting evolution. During a long time, its main goal was to provide efficient and accurate approximations of physical worlds in order to be able to simulate the reality using the available processing power: the main concern was the accuracy in creating 3D models and physically-based rendering. On illustration of such an approach is the “Rendering Equation” [Kaj86] that formalizes the recursive light propagation, and all its subsequent algorithms: with the increasing computational power and increasing understanding of the lighting phenomena modeled in this equation, we are now able to produce images that look so real!

This research approach is still very important in Computer Graphics, but a brand new trend has emerged. More and more, the research is now focusing on computing images (taking into account that their goal is to be seen) and on creative techniques: users are more and more put into the processing loop [Sei99]. This leads to an even more importance of interactivity (since it provides users with rapid feedbacks of their actions) and of comprehensible and simple parameters. One perfect example is the raising of sketch-based techniques for 3D modeling [IMT99]: 3D objects are defined by some of their visual characteristics such as silhouettes drawn by users as a set of 2D curves. Another example is the slow change from physically accurate global illumination¹ to plausible one: the resulting illumination has to look correct and fit the desired solution.

Another illustrative example is the still raising interest toward *expressive rendering*²: such approaches have first explored the simulation of traditional artistic techniques to develop a larger field due the extended possibilities of computers. They are more and more concerned with how users can tune comprehensible parameters to reach the final visual result that was intended. Interestingly, this change corresponds also to the administrative French naming of Computer Sciences: “Science and Techniques of Information and Communication”. We are more and focus on how we can present and transmit some information issued from 3D scenes in order to be more legible and cognitively efficient.

¹Physically accurate global illumination is often misnamed, in my opinion, as “photorealism”. Firstly, professional photographers often use optical and lighting tricks to create the desired pictures: this is no more the reality, it is already an artistic approach. Secondly, “photorealism” is another name for “hyperrealism”, an artistic movement in painting.

²This expression is more and more preferred to Non-Photorealistic Rendering (NPR): not being something is a too restrictive definition. Furthermore, this expression is often used to design more artistic approaches. But as seen in previous footnote, photorealism can also be artistic.

Personal scientific context

From my initial interest in accurate global illumination, I have followed a similar path. My PhD work [GD99, GD01, GD04b] was mostly focused on trying to provide simple and generic computational solutions for simulations. The intended control was mainly accuracy and not intuitiveness. The only user concern was the reduction of the number of parameters for these complex simulations. The assumption was that simpler and interactive approaches would result in more user-friendly controls. I have pursued this kind of development during my post-doctoral fellowship and its results in the creation of a generic and simple model for reflections [GH03].

Satisfying from a numerical and simulation point of view, or for a real-time rendering point of view, none of these approaches were convincing from a user point of view. Our global illumination solution requires an algorithmic and numeric knowledge of the simulation process, as knowledge of the simulated phenomena, in order to be able to control the solution. Furthermore, the results are only tied to physically plausible ones. Our simple model of reflection does not provide any comprehensible parameters but only pseudo-physical ones. None of them make it easy to reach the results that were expected by users, but results that are more or less physically correct. Once again, even if such an approach is important in Computer Graphics to understand the different physical constraints, it is not taking into account what I believe to be one of the specificities of this domain that is, creativity of users and their a-priori knowledge of their expected results.

Users need *levers to raise their virtual worlds*. In creative context, as an example, they need to be able to decide that the shape of a shadow has to be possibly non-geometrically feasible. When a large range of solutions is possible, they also need to be able to define themselves what the correct one is, and not based solely on purely automatic algorithms. Taking into account these constraints offers a new way of thinking the different problems and leads to new representations.

Goals and overview

This document is a unique opportunity for me to take a break, look back on the work done now since more than 10 years and to re-introduce it in this context. Each result has to be considered as an experimentation of my research approach. I have tried to explore as largely as possible different aspects of Computer Graphics while trying to preserve my non-expertise to evaluate their user-friendliness of the resulting approaches. This naivety has helped me in developing the presented solutions by taking a new look on classical problems.

Firstly, I will present some work done on geometric modeling, the first step of any modeling of 3D scenes. By trying to avoid the complexity of large software, we have focused our researches on 3D sketching. This reflection of user-intuitive 3D modeling leads to the definition of new surface-reconstruction schemes based on the set of user-drawn curves. We have also explored the use of such an approach in Virtual Reality Centers, where classical interaction devices are not sufficient.

The second step is in general the definition of shading properties of 3D objects. The second chapter details thus first, a new sketching and painting metaphor to define reflectance properties. Since appearance is also related to the geometric characteristics of 3D objects, I also present an approach that we have developed to analyze 3D shapes and to introduce back their features in a collection of expressive styles.

In the third and last chapter, I present an exploration of more global definitions of 3D scenes that is, the interactions between their different components and the light propagation. All the multiple inter-reflections and shadowing effects that occur have to be taken into account. This chapter illustrates how the reflection on a more user-friendly representation can result in a more generic solution.

The presented representation offers a geometrically robustness that is well-suited for a number of applications, such as interactive rendering, streaming of 3D scenes, and caching structure for stochastic integration.

To conclude this document, I will not only review all the contributions, but I will detail some research directions that can result from the presented solutions. From a general overview, this presentation will also explore some specific applicative contexts.

Sketching 3D Models

The creation of 3D images relies first on the definition of a collection of 3D objects and on their relative positions. Inherited from its mathematical history, 3D modeling is issued from CAGD and thus based on 3D primitives such as meshes, parametric and implicit surfaces. Due to the resulting large number of tools for each geometrical approach, the interface of current 3D modelers may be extremely complex and intimidating for non-expert users. Creating prototypes of 3D models may thus become a slow and painful task.

Since more than ten years now, new approaches for 3D modeling have been developed, considering our human ability to quickly describe the general shape of a 3D object via simple drawings. Before considering any lighting, we will thus take a tour into the definition of 3D worlds from their geometric aspect, using user-friendly interaction named *Sketching* for 3D modeling.

We focus on two aspects of free-form modeling. The first one is the creation of objects **more complex** than classical blob-like ones, and with **higher genus**. This development leads also to some quality improvements over previous mesh-based reconstructions. The second one is the **interaction** aspect to either **expose comprehensibly** all the reconstruction steps and parameters, and to intuitively position the different object's components in the context of **reality centers**. Apart from the Section 1.2.2 and Section 1.2.3, the results presented in this chapter have been validated by the following publications: [LGS06a, LGS07b, LG07, ZHG08].

1.1 Context and Previous Work

Artists use many techniques to suggest the 3D object shape, techniques such as characteristic lines (or contour lines) or shaded areas. Since drawing is a familiar task for a lot of people, “sketching” has been introduced as a natural alternative to conventional 3D modeling (a very good overview has been published by Olsen et al. [OSSJ09]). Existing sketching approaches can be divided into three categories: the line-and-stroke approach, the painting-or-shading approach, and the curve-and-gesture approach.

Lines and Strokes

The principle of the line-and-stroke approach is to infer a 3D volume from characteristic lines issued directly from user-drawings [EHBE97, VTMS04]. Interactivity [Pug92, SC04] is even possible with

such a solution. When ambiguities occur, they may be removed by user-selection of a suitable model into a list of alternative reconstructions [PBJ⁺04, FFJ04]. Most of these approaches are limited to polyhedral models, pre-defined shapes or parameterized objects [YSv05]. Recent methods [CMZ⁺99, TBSR04] result in complex 3D lines, but final models are still limited to a collection of oriented planar curves. Some stroke-based approaches [TDM99, BCD01] allow the creation of real 3D curves, but such solutions do not result in a 3D object, but still in a collection of strokes.

Painting or Shading

Shading or painting seems to be a natural approach to describe a shape, since part of its perception is based on its shading. Based on this assumption, the painting-or-shading approaches allow the generation of highly detailed models. As the shading is in general dependent on normals (i.e., local surface variations), it seems natural to use it to provide a visual control of the shape. Extending the work of Williams [Wil90], Overveld introduced a shape modeling method by painting the surface gradient [van96]. Some constraints guarantee that a solution exists, but they disallow the use of classical 2D paint programs and thus reduce users' freedom. An original approach to edit shape by using shading information was introduced by Rushmeier et al. [RGB⁺03]. Their method was used to restore the original shape by using a shading image associated with the region to be edited. This approach is convenient to edit small parts of an existing 3D shape, but it is not adapted to create a new shape without an initial 3D model. Moreover, since it makes use of only one light direction a time, this approach is more restricted and less robust than solutions based on multiple ones.

More recently, Kerautret et al. [KGB05] has shown that height-fields may be reconstructed using shadings drawn from multiple light directions, but their approach was limited to height-fields and was not interactive. Wu et al. [WTBS07] and Ng et al. [NWT07] have shown how a normal field and thus a surface may be reconstructed from sparse constraints on normals, but were less intuitive than the original Lumo [Joh02] where the constraints were discontinuity curves. Nevertheless, shading-based solutions have been successfully extended to edit existing shapes at interactive frame-rate [GZ08], using laplacian constraints for shape and normal changes. But they are limited to local shape edition.

Curves and Gestures

Currently, the most powerful techniques have been based on the curve-and-gesture approach. These techniques allow users to create a large variety of free-form shapes [ZHH96, IMT99, ONNI03] by using a gesture grammar which converts some drawn curves into a corresponding modeling operation: extrusion, deformation, cutting, etc.

Either variational surfaces [KHR02, ŽS03], implicit surfaces [SWSJ05] or convolution surfaces [ABCG05] have been used by curve-and-gesture approaches. This has the advantage to generate smooth surfaces, but also emphasizes a "blobby" aspect for the resulting shapes. To reduce this blobby aspect, Tai et al. [TZF04] have proposed to use a profile curve, defined in polar coordinates. Profile curves have been used previously in a sketching environment limited to generalized cylinders [GH98]. One nice property of implicit surfaces for the inferred geometry is that the resulting shapes are easily merged by using classical CSG operators. On the other hand, the main drawback is that some expensive tessellation step has to be employed to convert the surface into a set of triangles that are sent to the graphics hardware, and complex shapes require a lot of CSG operations.

But for some modeling operations, as for rendering, even implicit surfaces need to be locally sampled (e.g., [BPCB08]). Furthermore, fitting some constraints like silhouette curves may be time-consuming (e.g., [TZF04, ABCG05]). On the other side, direct rendering is possible with meshes,

or more generally with sampled surfaces, and such surfaces offer in general more flexibility: recent researches on point-based surfaces (e.g., [KHR02, GG07]) have shown that a sampled surface is easily interpreted as an implicit surface when required. In our work, we thus rely on these surfaces as our main representation.

One of the most impressive reconstruction techniques to create 3D models based on sketched profile curves was presented by Cherlin et al. [CSSJ05]. Unfortunately, it is not possible to create topological complex objects (i.e., with branches) and it would require the use of a skeleton. Besides, for one object, users have to sketch two disconnected silhouettes. This could be confusing since it seems more obvious to sketch only one closed silhouette.

Skeleton Extraction

The skeleton extraction introduced in Teddy [IMT99] depends on an initial triangulation of the approximated silhouette, and is prone to artefacts leading to a low quality mesh [IH03]. Even when using implicit surfaces [AJ03, TZF04, ŽS03, ABCG04], the final quality depends on the quality of the skeleton.

In *ConvMo* [TZF04], the segment-based skeleton is improved by removing some vertexes, but it still depends on the initial silhouette sampling. Furthermore, the profile is limited to polar coordinates. Alexe et al. [ABCG05] use an image processing technique for the skeleton creation: the resulting skeleton depends only on image resolution. But it contains not only segments and but also polygons: suitable for convolution surfaces, this is not suitable for profile-based techniques that require curve or segment-based skeletons.

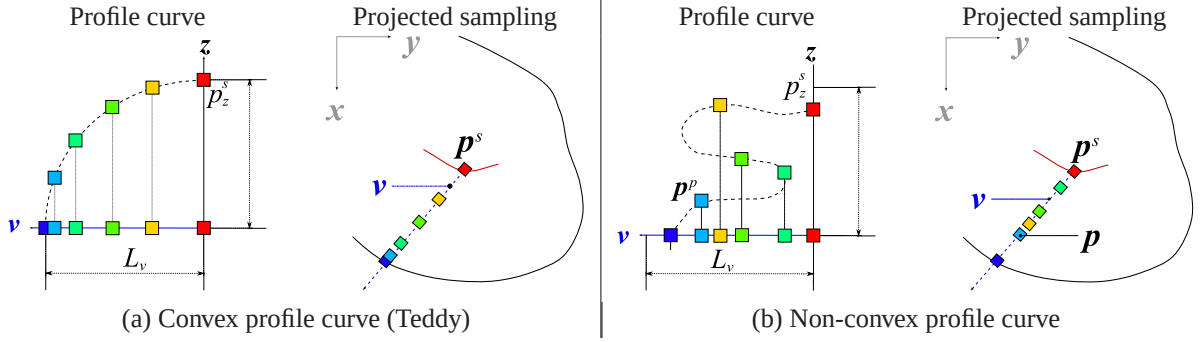


Fig. 1.1: Differences for surface elevation between Teddy and our approach: p_z^s is the silhouette elevation, v the direction of the radial edge and L_v its length. In Teddy (left), radial edges from skeleton to silhouette are sampled and the samples are elevated. In our approach (right), the profile-curve is sampled, and the position of these samples are adjusted for each radial edge, based on the local maximum elevation and edge length. The projection of the resulting 3D vertices on their corresponding radial edge is no more guaranteed to be ordered.

1.2 Profile Curves for Enhanced Free-Form Modeling

Based on these observations, we have designed a sketching environment on the following criteria. First, our system directly infers a mesh from a set of curve-and-gesture elements. In addition to speed for inferring a shape, it is easier to guarantee with a mesh that the silhouette of the resulting objects will match the drawn one: with implicit surfaces such as convolution surfaces, a costly optimization step is required [TZF04]. Moreover, using a mesh would offer the possibility to locally adapt the sampling according to the local density of details [BPCB08]. Second, our system extends the grammar defined in Teddy [IMT99] by creating a profile curve for the geometric model that is edited either globally or locally.

Finally, we want to expose to the users all the reconstruction steps and the internal representations used for this reconstruction. We believe that this exposition would provide users with more understanding of the underlying modeling process and with more flexibility.

1.2.1 Profile-Based Approach

Our approach is based on the Teddy system [IMT99], and thus uses a similar four-step process:

1. Users sketch a silhouette curve then this curve is sampled.
2. From the silhouette curve, a skeleton is extracted.
3. Along the skeleton, an elevation distance is computed, as the average distance to the closest silhouette sections. Radial Edges (segments from skeleton to silhouette) are also created (see Section 1.2.3).
4. During the last step (the elevation step), we create 3D points, along each radial edge, corresponding to a profile curve. Users may further manipulate this profile.

Compared to the original Teddy implementation, our system proposes some improvements: the elevation step is modified, in order to account for non-convex profile curves; we also introduce, in the following sections, a new and smoother skeleton extraction, the corresponding estimation of radial edges, and the final surface reconstruction.

Elevation

With Teddy, mesh vertices are created by sampling the radial edges on the silhouette plane $\mathbf{xy}$ and elevating them according to a circular profile curve (see Figure 1.1-(a)). Unfortunately, this only work for convex profile curves (i.e., profile curves expressed as a height-field). For our approach, we want to create more complex shapes by defining more complex profile-curves.

To use non-convex profile curves, we propose to compute a set of sample points directly on the profile curve $\mathbf{p}^p = (p_v^p, p_z^p)$, and use these samples along each radial edge to obtain the position of the corresponding mesh vertices $\mathbf{p} = (p_x, p_y, p_z)$. Note that we rescale $\mathbf{p}^p$ to fit the following constraint $p_v^p \leq 1$. This constraint guarantees that the final object will stay inside the silhouette, and that it does not contain self-occlusion, apart if the silhouette contains some. Indeed, coordinates p_x and p_y are computed as

$$p_x = p_x^s + v_x L_v p_v^p \quad \text{and} \quad p_y = p_y^s + v_y L_v p_v^p$$

where $\mathbf{p}^s$, using the notation of Figure 1.1, is the skeleton point from which this edge is issued. The final elevation of each vertex is thus a simple scaling given by

$$p_z = p_z^s p_z^p,$$

where p_z^s is the corresponding skeleton elevation.

1.2.2 Skeleton Extraction

A smooth skeleton is critical for the generation of high-quality objects. Furthermore, in order to expose the different steps of the 3D model generation, each operation has to be robust. We have first experimented a skeleton extraction from an implicit 2D function [LG07]. The basic idea is that the skeleton follows a line of local minima of the implicit function. Even if the skeletons are smooth, the approach is not robust: it was based on a time consuming gradient descent.

We thus improve it with a hybrid method based on a binary image analysis and still, a 2D variational implicit function (both of them reconstructed from the sketched silhouette). On one side, thinning (e.g., [MBPL99, ZQN95]) generate skeletons that preserve the shape topologies [Cou06]. However, they are noise-sensitive and usually yield unwanted edges. On the other hand, skeletons extracted from implicit functions are smooth [LG07, MWO03]. Unfortunately, depending on the implicit function, this approach may smooth out some features. In our approach, a thinning algorithm, which preserve the shape topology, is used to find the skeleton seeds (i.e., the vanishing points of the distance function to the silhouette), and an implicit function is used to generate the final smooth skeleton between the seeds.

Skeleton Seeds

The first step consists in the construction of a binary image from the sketched silhouette. On this image, we apply a thinning process to find skeleton seeds. To speed-up the process, we use a parallel approach [ZQN95]. At each step, any pixel of the image boundary is potentially removed (contrary to a sequential thinning that only removes pixels in a given direction). However, a retrieval mask has to be applied in order to preserve non-simple pixels (corresponding to junction between the different parts: they are topologically important).

As shown in [Cou06], the algorithm guarantees that the shape topology is preserved. For each boundary pixel of the binary image, 8 thinning masks (see Figure 1.2-(a)) are applied. In order to ensure a 1-pixel width skeleton, we introduce two retrieval masks (see Figure 1.2-(b)) in the original

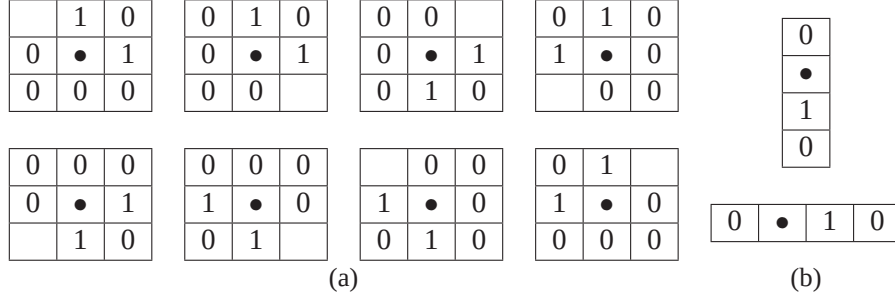


Fig. 1.2: (a) Thinning masks for the technique of Ng et al. [ZQN95]. • is the pixel currently treated and empty cell means that the value is 0 or 1. (b) Restoring masks added to ensure to have a 1-pixel width.

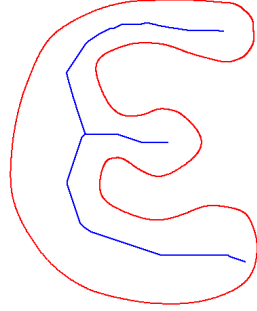


Fig. 1.3: Skeleton from thinning.

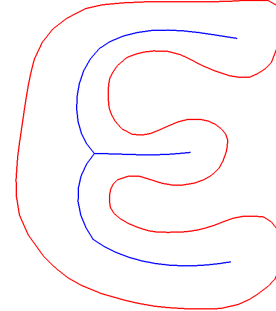


Fig. 1.4: Smoothed skeleton.

approach [ZQN95]. If any of the thinning masks and none of the retrieval ones are matched, the pixel can be removed while preserving the topology. The resulting skeletons (see Figure 1.3) are far from being smooth but, they are used to determine the seeds (points that embed the core shape topology). These seeds are easily identified using the pixel connectivity on the resulting skeleton:

- 1 neighboring pixel: the pixel is a **final seed**
- 2 neighboring pixels: the pixel is not a seed
- more than two neighboring pixels: the pixel is a **junction seed**

Skeleton Edge Generation

Starting from the seeds, the skeleton edges are curves that connect two neighboring seeds. Finding these neighboring relationships between seeds is simple, based on the skeleton generated by thinning. To smooth the skeleton edges, a mass-spring system defined by Equation 1.1 is used for each skeleton point i from the skeleton edges:

$$\mathbf{p}_i = \mathbf{p}_i - \nabla F_i + \sum_j \mathbf{r}_{ij} \left(\delta \frac{\phi - |\mathbf{r}_{ij}|}{\phi} \right) \quad (1.1)$$

where $\mathbf{p}_i$ is the position of skeleton point i , ∇F_i is the gradient of implicit function F at position $\mathbf{p}_i$, $\mathbf{r}_{ij}$ the vector between the point i and its neighbors j , ϕ the radius of the particles and δ the global length of the springs¹.

Since seed positions are fixed, only skeleton points belonging to the skeleton edges are moving accordingly to Equation 1.1. Since the mass-spring system is influenced by the implicit function, the

¹in our current implementation, $\delta = 2\phi$ and ϕ is defined as $\phi = \Psi/k$ where Ψ is the diagonal of the object bounding box and k is a user-defined scaling factor of ϕ

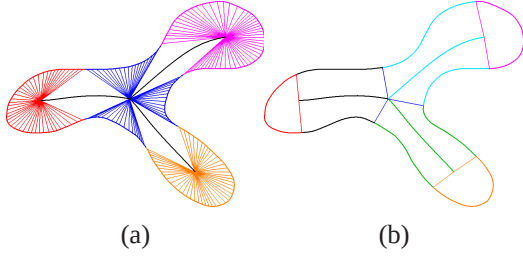


Fig. 1.5: (a) Radial edges and their dependency on the seeds. (b) Silhouette segmentation based on the previous radial edges. The colors show the correspondence between the skeleton and silhouette components.

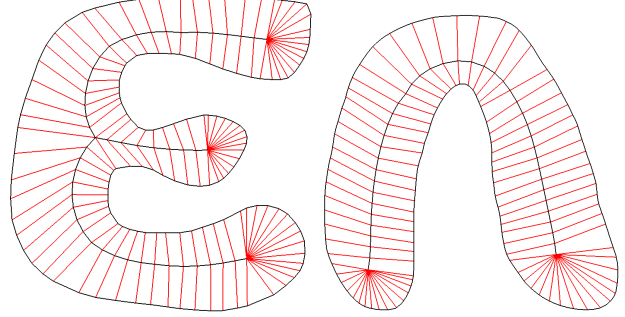


Fig. 1.6: New radial edges from our approach.

skeleton edges tend to follow the lines of local minima. Thus, skeletons generated by our technique (see Figure 1.4) are smooth while preserving the topology.

1.2.3 Improved Surface Reconstruction

We generalize the work of Cherlin et al. [CSSJ05] to any closed silhouette. To this purpose, we use the skeleton generated with our approach (see Section 1.2.2) to segment the silhouette into components on which this technique [CSSJ05] can be used for inferring the 3D model.

Silhouette Segmentation

The silhouette segmentation is based on radial edges (edges that connect a silhouette point to a skeleton point). The initial radial edges are connected to their closest seeds. The set of radial edges connected to a seed correspond to the silhouette region that depends on it (see Figure 1.5-(a)). According to the seed type (junction or final), a different algorithm is used to select a subset of radial edges which support the final segmentation (see Figure 1.5-(b)). The silhouette section that connects two selected edges is called a **silhouette edge**.

Final seed: these seeds have only one neighbor but two radial edges are needed to segment the silhouette region. We keep only the two radial edges that are the most orthogonal to the skeleton edge connected to this seed (see Figure 1.5-(b)).

Junction seed: the number of required radial edges is equal to its number of neighboring seeds. Thus, the seed is simply connected to the closer silhouette points for each seed-dependent region of the silhouette (see Figure 1.5-(b)).

Inferring a 3D Surface

Starting from this set of radial edges and as in [CSSJ05], each skeleton and silhouette edges are smoothly reconstructed as *splines*. Indeed, as illustrated in Figure 1.5-(b), each silhouette edge corresponds to either a skeleton edge or a final skeleton seed. Each skeleton edge is related to 2 silhouette edges.

To compute a final set of radial edges (see Figure 1.6), the 3 edges are sampled with the same number of points. Then, it is sufficient to connect the skeleton points of the skeleton edge to their 2 corresponding silhouette points. Finally, for the silhouette edges that are related to a final skeleton seed, the radial edges are created by linking the silhouette points to the seed.

Then, the radial edges are sampled and these samples are simply triangulated with their immediate neighbors on the silhouette plane and are displaced according to the profile (see Section 1.2.1).

1.2.4 Multi-View Interface

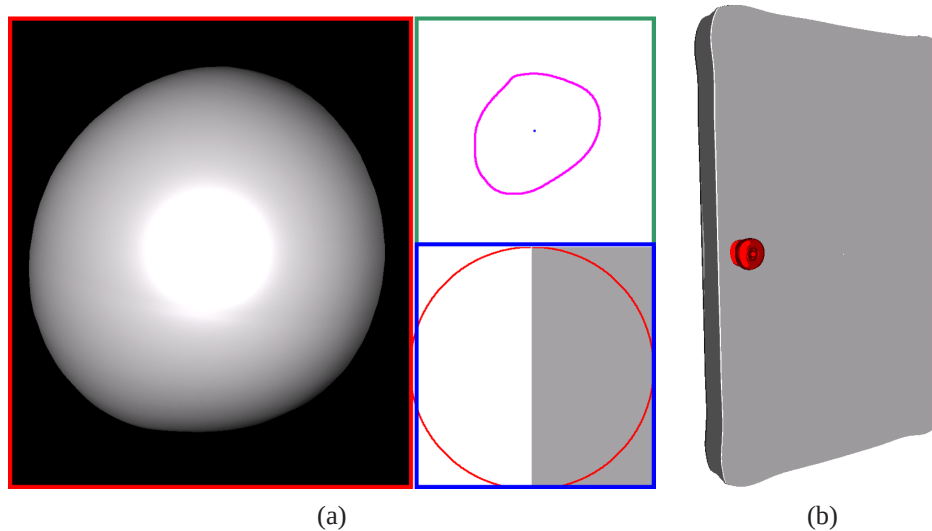


Fig. 1.7: (a) Three different views are used in our system: the 3D view (left frame) that shows the 3D object under construction, the skeleton view (upper-right frame) which shows the outline of the object and its skeleton and, the profile view (bottom-right frame) that shows the profile currently used to create the object. (b) A door created with two objects. Their relative 3D positioning was done in the 3D view.

Since this approach provides users with a larger range of possible resulting shapes, it is even more essential to integrate all the actions in a simple interface, with comprehensible feedback. Our prototype, shown in Figure 1.7, is based on three different views: the 3D view (left frame), the skeleton view (upper-right frame) and the profile view (bottom-right frame). Each view is specialized into a component of the modeling process and defines its own sketching interactions. For easier undo or redo actions, we complete the profile view by a complementary one, allowing the selection of an existing profile curve.

3D View

The 3D view is very similar to the standard one-view of other 3D sketching systems, and is mainly used to perform three tasks: (i) the visualization of 3D objects, (ii) the positioning of these 3D objects (either by moving them - see Figure 1.7-(b) - or by defining a new 2D plane for the silhouette curve) and, (iii) 3D editing of objects. All the Teddy-like operations such as cutting, extrusion or fusion might be implemented in this last view.

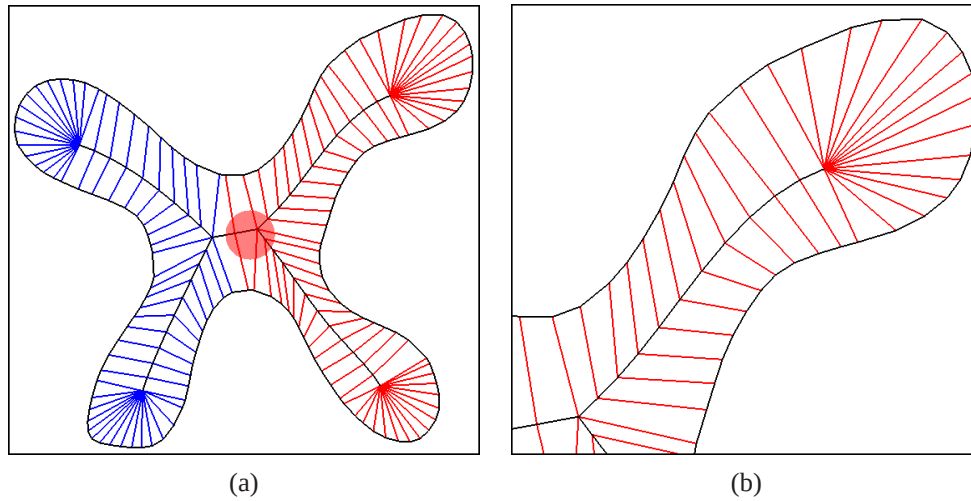


Fig. 1.8: (a) Using the selection tool to select a set of radial edges. (b) The zoom tool focus on a part of the silhouette/skeleton.

Skeleton View

This view shows the silhouette shape as well as its skeleton, and is a very good illustration of what we intend to do by exposing all the underlying process and representation. In sketching approaches, the direct selection in the 3D view may be difficult, due to the 3D projection on the screen. With multi-view systems, this selection is performed in the 2D skeleton view. Since, there is no projection from a 2D sketch to a 3D space, the result is much easier to predict.

Profile View

All the operations related to the profile curve are defined on the profile view which is the last view provided by our system. More precisely, users only have to draw a 2D sketch to define a new profile. In our current implementation, only half of the curve has to be drawn and the other half is reconstructed by symmetry. Besides, since users may want to create a 3D blobby object by only sketching a silhouette curve, we have designed a default circular profile curve used to initialize of our system session.

Once the profile curve has been generated, it is then sampled and the sample points are transferred to the 3D view to be applied on the current selected area. If no local area of the object has been selected, the new profile curve is applied on the whole object.

Profile Selection

This view keeps track of and manipulates the defined profiles. All the sketched profiles are saved with the selected radial edges on which they were applied. For each profile, users have 3 choices²: (1) the radial edges on which the profile has been applied are selected, (2) the profile is applied on the current selected radial edges or, (3) the profile is re-applied on its defined selected radial edges to restore a previous state. With a scribbling sketch on a profile, users simple delete it. This view is not

²the choice is currently done according to the mouse buttons, since this interface is currently designed for a desktop workstation. It might be easily replaced by any gesture/sketch interactions for other platform

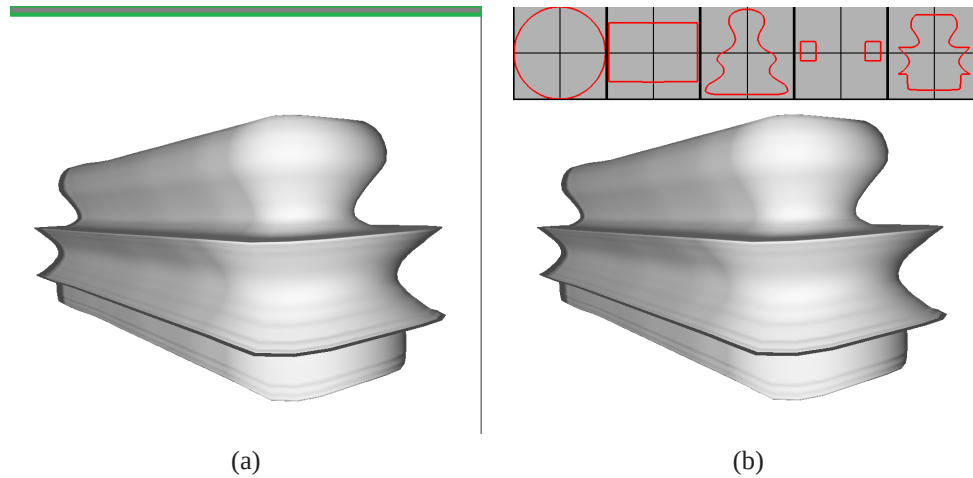


Fig. 1.9: (a) The history of profile used in the modeling session is in general hidden and only revealed window (b) when requested.

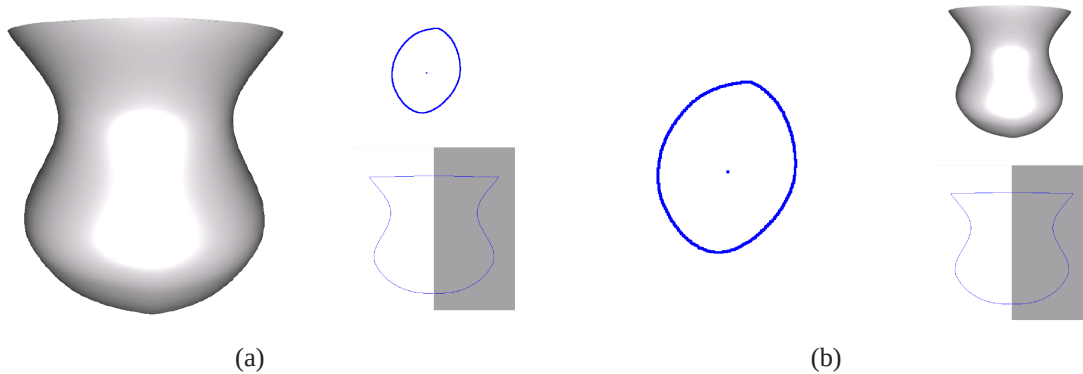


Fig. 1.10: (a) Default setting of the system with the 3D view as the principal view. (b) Swapped between skeleton view and 3D view.

always shown since it is only useful when radial edges are selected: the view appears when selected (cf. Figure 1.9-(b)) and is hidden as soon we leave it (cf. Figure 1.9-(a)).

Ordering the Multiple Views

Finally, it is possible to change the view layout (apart from the profile selection). Since the 3D view is not necessarily the most important view for a specific operation, users may want to use the skeleton or the profile view in place of the principal view. We thus offer a straightforward sketch-based interface to swap the position of two different views: this is done by starting a stroke in one view, and finishing it in the other one.

1.2.5 Discussions

We have presented new techniques to improve the quality of models generated using sketch-based free-form modeling systems: we have introduced a new skeleton extraction, a new scheme to create the set of internal edges which aim is to limit the creation of tiny and uneven triangles. Indeed, better internal edges lead to the creation of better meshes for the final object. Finally, we have presented

how users manipulate the profile curves to create more complex objects.

For the modeling aspect, the main problem remains the transition between two profiles that is not well-defined. Furthermore, the smoothness of the reconstruction is not guaranteed when the radial edges are not close to orthogonal to the corresponding skeleton edge (this is also problem of the original technique of Cherlin et al. [CSSJ05]). We need to investigate how to insure this orthogonality, and how to find a better radial edge generation near the junction skeleton-seeds.

On the interaction side, the only validation is our own experience. Even if we believe that such an approach could lead to more flexible modeling interactions, a user-study comparing the efficiency to this approach to existing ones is required.

1.3 3D Positioning for Large displays

Similar to our system, most of the previous ones are based on a classic 2D *pen-and-paper* context, similar to the use of a drawing tablet. In general (e.g., [IMT99, AJ03, TZF04]), the final model is obtained by combining different components resulting from 2D sketches on different planes. However, combining these different components requires switching between 2D sketching interactions and 3D manipulations (mainly positioning and rotation). This leads to our development of an integrated approach for these two modes, resulting in more comprehensible changes in-between them.

The main goal is to provide an comprehensible manipulation environment for 3D design in virtual reality centers. In these environments, several users might collaborate and share ideas for creating 3D prototypes in front of a large display. For easier interactions, we propose to use an interaction device that provides a direct mapping between its physical orientation and the relative orientation of a 3D object on the screen. This direct mapping guarantees that any position is easily visually and physically recovered by any user. It leads also to the introduction of the 3D metaphor of virtual 3D paper sheet similar to the 3D canvas introduced by Dorsey et al. [DXS⁺07].

1.3.1 Previous Work

One solution to the 3D positioning problem is to remove the restriction of sketching only 2D curves, by providing an interaction to directly draw 3D curves with a tracked device. Lapidès et al. [LSSS06] use a tablet mounted on a support that can be translated vertically. One hand is used to draw a 2D curve on the tablet and the other one to add the third dimension by a vertical translation of the tablet. Unfortunately, a good 3D visualization system is required for accurate 3D drawing; otherwise, these approaches would result in undesired solutions due to the differences between the visualization system and our spatial representation of a 3D world. Furthermore, these techniques demand acute coordination abilities, involving both hands.

It is usually easier to have a set of reference planes [TBSR04] or surfaces [GBK⁺02] to draw on. Tsang et al. [TBSR04] use a set on planes oriented along the main axes. Markosian et al. [MCCH99] use the projection of the 3D curves on the image plane and its projection on a reference plane (called a “shadow curve”) for inferring the 3D shape. The sequence of interactions is inverted by Grossman et al. [GBK⁺02] to extend the reference plane to a reference surface generated by the “shadow curve”. They use a “Tape Drawing” [BFKB99] approach for curve sketching.

More recently, approaches based on multiple reference planes have been extended by Dorsey et al. [DXS⁺07] for architectural design and analysis. Their system uses strokes and planar “canvases” as basic primitives like in a traditional sketchbook (one sketch for each canvas) but do not provide true free-form models.

For an improved kinesthetic feedback, some tracked two-handed interactions have been introduced. Sachs et al. [SRS91] use two Polemus-tracked devices: a palette as the reference plane and a stylus of the sketching part. Thanks to the palette, 3D positioning of the 2D curve or 3D object is easily achieved. Schroering et al. [SGP03] use the same approach with camera-tracked devices: a board as reference plane and a laser pointer as a stylus. Unfortunately, as for any hand-held device, this may be physically tiring for users. Furthermore, when multiple users are involved in the modeling process, the direct matching between the devices and the 3D positioning is lost anytime the devices are released, put back at their original position, or passed to another collaborator.

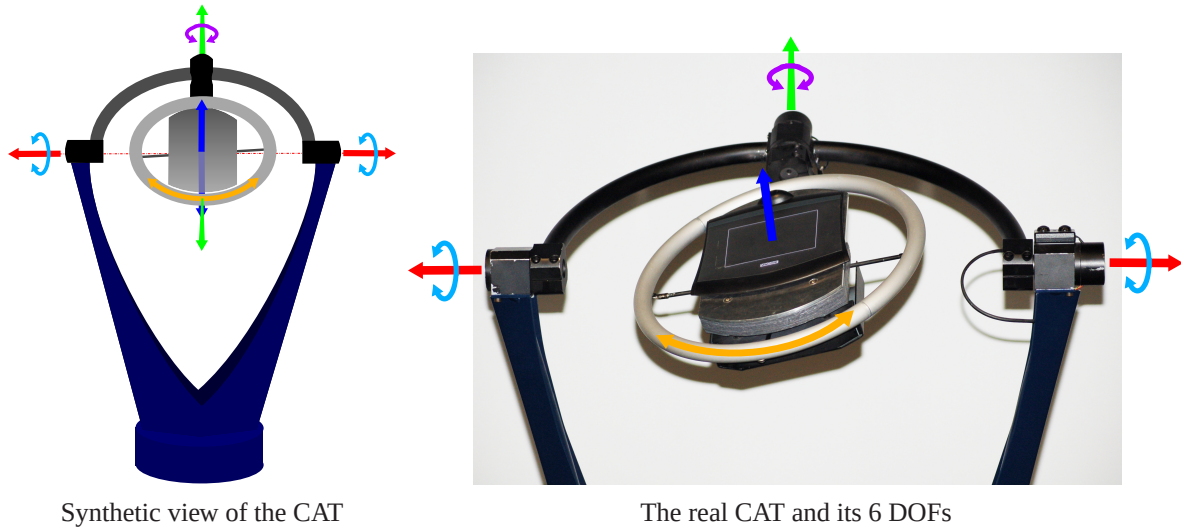


Fig. 1.11: Presenting the CAT and all 6 DOFs. The table can be rotated along all the 3 axes. Translations are detected by user's pressure along the 3 axes. Note the drawing tablet mounted on the table.

1.3.2 System Overview

We introduce a virtual 3D paper sheet metaphor in our sketching system. We assume that users draw different sketches on different reference planes and combine them together in order to create the final object [DXS⁺07]. For the combination, we extend the classical 2D desk to a 3D space.

Among the recent interaction devices, the CAT [HG04] (cf. Figure 1.11), a 6 DOFs interactor, provides us with a solution to the problems of 3D positioning. Compared to other devices, the CAT favors an unconstrained interaction since users do not have to hold anything. Furthermore with the CAT, rotations are directly controlled by an isotonic sensing mode while (infinite) translations are controlled by an isometric sensing mode. Thanks to these features, it allows intuitive manipulations of 3D objects: the orientation of the table directly corresponds to the orientation of a plane in 3D space. Most similar devices like the SpaceMouse or 3D Mouse do not share such convenient properties. Our CAT-based solution may thus simplify the orientation of the reference plane required for sketching. Furthermore, on one side, a translation is in theory an unbounded transformation and thus has to be relative. On the other side, a rotation is bounded and thus has to be absolute. These interactions are naturally offered by the CAT.

Moreover, a tablet fixed on the tabletop of the CAT is used for 2D interaction in our system. To track the original plane where the 2D sketches are drawn, this tablet is directly mapped to a 3D virtual paper sheet. Thanks to the 6 DOF available on the CAT, this virtual sheet, and thus users, is still immersed in 3D environment. Therefore, all the interaction sequences have thus to be carefully designed in order to use this 6 + 2 DOFs and will be described in the next section. With only one device, we access to all the required DOFs.

Virtual 3D Paper Sheet Metaphor

A sketching system is naturally based on a paper-and-pen metaphor. With the CAT, a tablet mounted on the top is used as a reference plane, and thus directly associated to a 3D virtual paper sheet. The main problem for editing the different components of the object is thus to correctly position a set of reference planes in 3D (similar to the canvas in the work of Dorsey et al. [DXS⁺07]). Two main

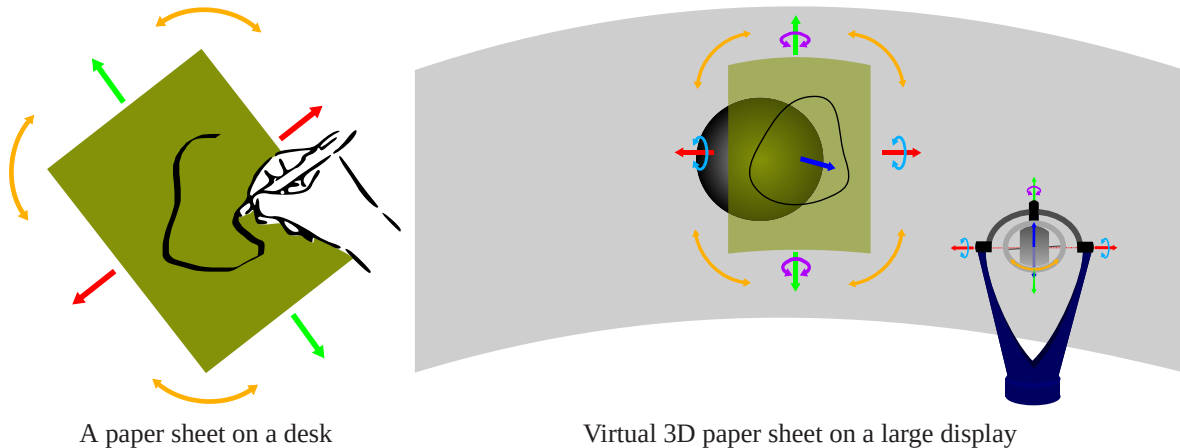


Fig. 1.12: Movement of a real paper sheet on a desk (left) and the corresponding mapping between the rotations and the translations of the CAT and the movements of the virtual 3D paper sheet (right).

approaches may be used for this positioning: the traditional manipulation of the scene/object, or the manipulation of a virtual 3D paper sheet.

For the first one, users have to position the 3D scene/object relatively to a fixed 3D paper sheet. Unfortunately, it might be difficult to map the arbitrary shape of the resulting 3D object to the planar tabletop: the choice of how the orientation of any object is mapped to the tabletop orientation is arbitrary without any visual cues. As an example, how can we attach a spherical object to a rectangular tabletop?

On the CAT, the paper sheet can be directly associated with the tablet mounted on the moving table. We thus naturally decide to use this second approach: the positioning of a reference plane in 3D. The table orientation directly corresponds to the orientation of this 3D paper sheet (see Figure 1.12). Users just feel as they were in front of a desk, drawing multiple sketches on multiple paper sheets, and assembling them together: rotations of the tablet and translations are directly interpreted as translation and rotations of a virtual 3D paper sheet.

1.3.3 Modeling Sessions

The modeling session is generally broken down in two main steps (see Figure 1.13). During the first one, users are moving the supporting plane relatively to the 3D scene. Once this virtual 3D paper sheet is correctly positioned, users have to change to sketching mode. Thanks to the CAT, these two steps are performed using a single device. When positioning the virtual 3D paper sheet relatively to the scene, it is drawn as a transparent plane.

In positioning mode, only the virtual plane is user-controlled. The direct mapping between the table orientation and the virtual 3D paper sheet simplifies this process.

In sketching mode, the virtual 3D paper sheet and the 3D scene are linked together: any translation or rotation of the CAT is directly applied to the scene and the virtual 3D paper sheet. This has two main advantages. First, if the tablet is moving during the sketching session, the relative positioning is not lost between the plane and the scene. Second, for a user point of view, it is easier to draw with the table in a horizontal position. This horizontal configuration is definitively not guaranteed after all the rotations occurring during the positioning of the virtual 3D paper sheet. To finalize the transition to the sketching mode, users have thus to manually rotate back in this position and – since the object and the virtual plane are linked together – the whole scene (i.e., the 3D object and the virtual plane) is

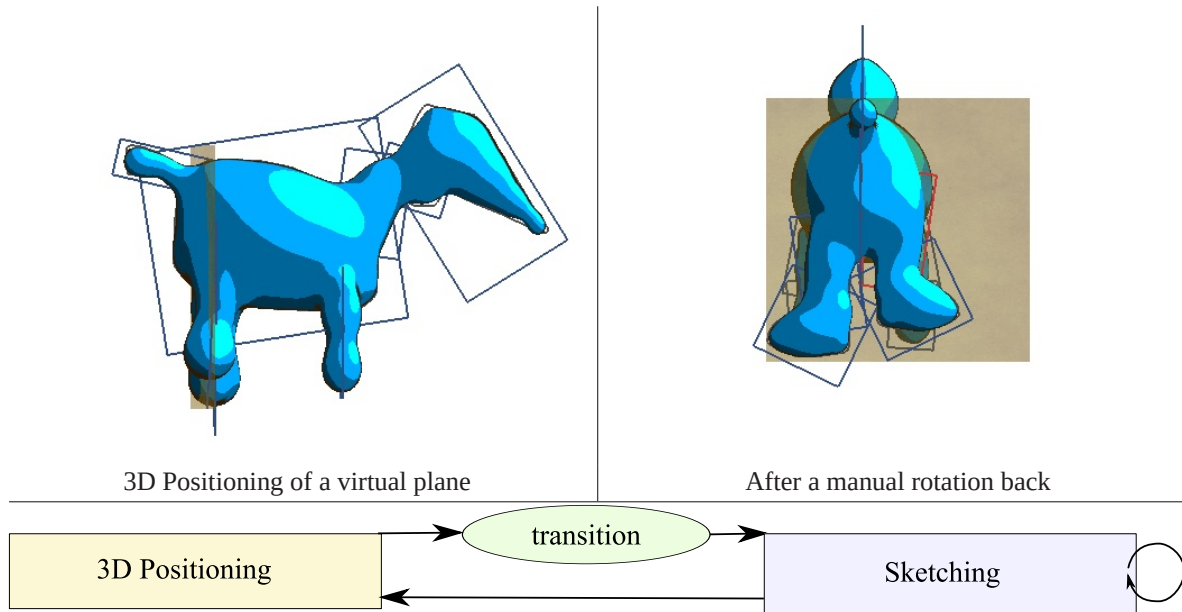


Fig. 1.13: Positioning of the virtual 3D paper sheet. The two modeling steps (upper images) and the associated interaction graph (lower image). During the 3D positioning mode, only the virtual plane is manipulated. During the sketching mode, the scene and the virtual 3D paper sheet are linked together. After the transition from positioning to sketching mode, users are allowed to rotate the virtual plane and the CAT to a horizontal configuration, more comfortable for drawing.

rotated in order to move back also the virtual 3D paper sheet in a position closer to the initial one.

Note that, during the transition from the positioning mode to the sketching mode, the whole scene is also translated in order to move the center of the virtual 3D paper sheet at the center of the screen. This guarantees that, if the inverse rotation is applied to the horizontal position, the virtual 3D paper sheet will be back in the initial orientation and also centered on the screen, in a configuration similar to the original one (see upper-left image in Figure 1.13).

Once the silhouette has been sketched, we save its association with the reference plane. A reference plane is defined by a center $\mathbf{c}$ (the center of the silhouette) and a normal $\mathbf{n}$ (defined by the orientation of the virtual 3D paper sheet). We store also the corresponding 2D bounding box for display.

Editing is a bit more complex. In the ideal case, when adding a new component to the final object, the previous 2-step approach is sufficient. To increase the users' freedom, and the possibility of multi-users interaction, editing also needs to be supported. For most of the editing tasks, we need to retrieve the supporting plane corresponding to the component that we want to modify. Once again, thanks the CAT, this is easily done. Users move the virtual plane in the 3D scene, and when it coarsely corresponds to the target plane, this one is selected. Users may thus want to return the table to the initial position (horizontal) and performs the 2D operations. Similarly to the modeling task, the whole scene is translated during this change of mode, and afterward rotated if users want to move back the virtual plane back to its original position.

The selection of the closest reference plane is done based on the plane orientation (i.e., normal $\mathbf{n}$) and its center $\mathbf{c}$. To assist users in selecting the correct reference plane corresponding to the object's component she aims to modify, we use color cues. A color is associated to each component. This color is used to display the reference plane and its corresponding surface (see Figure 1.14). We select a set of perceptually different colors, excluding red, as we use this color to highlight the current closest

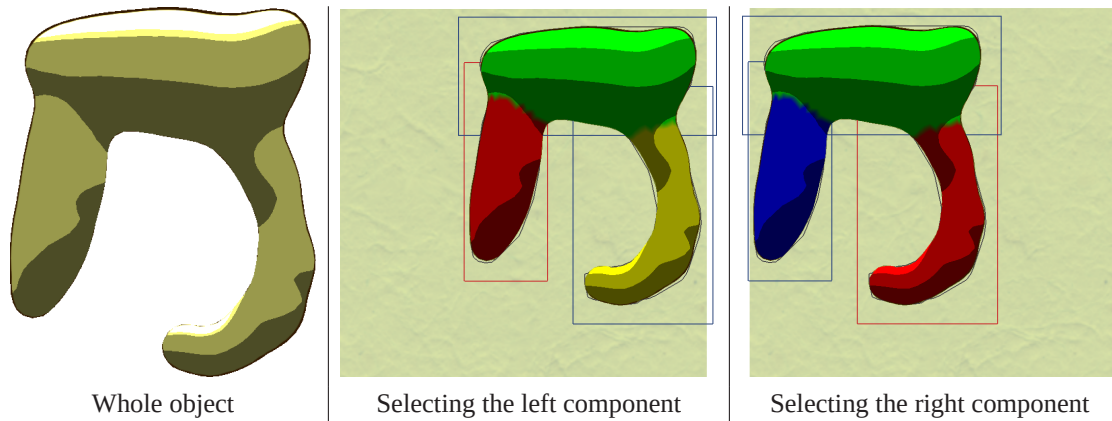


Fig. 1.14: Selection of a reference plane. In edit mode, each component has its own color and the selected component is highlighted in red. We take the closest one to the center of the virtual 3D paper sheet.

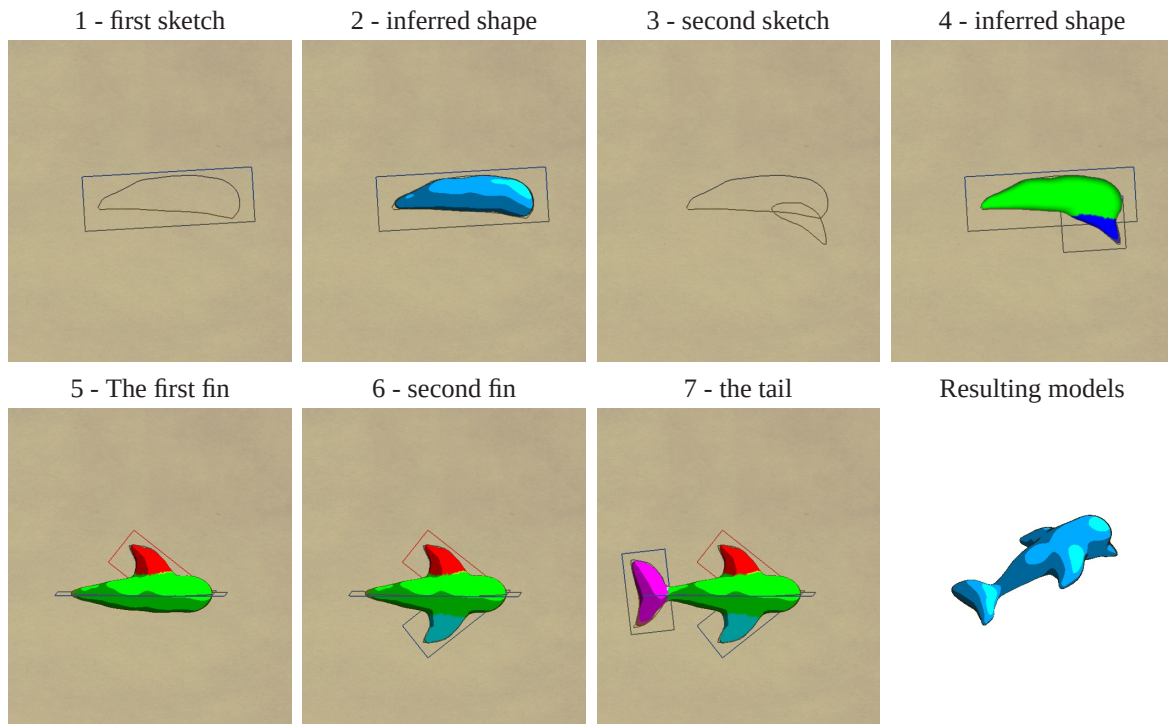


Fig. 1.15: Modeling a dolphin. The first four steps are done with nearly coplanar 3D paper sheet. During the transition from step 4 to step 5, the CAT has been used to position the virtual plane orthogonally to dolphin's body, and rotated back as explained in Section 1.3.3. The steps 5, 6 and 7 are now performed on the same plane

plane.

1.3.4 Results

To illustrate a typical modeling session with the presented approach, we review the steps involved in creating the shape of the dolphin in Figure 1.15. Users first draw the main body of the dolphin using two components (steps 1 to 4). To rotate the plane into a position orthogonal to the main body, the

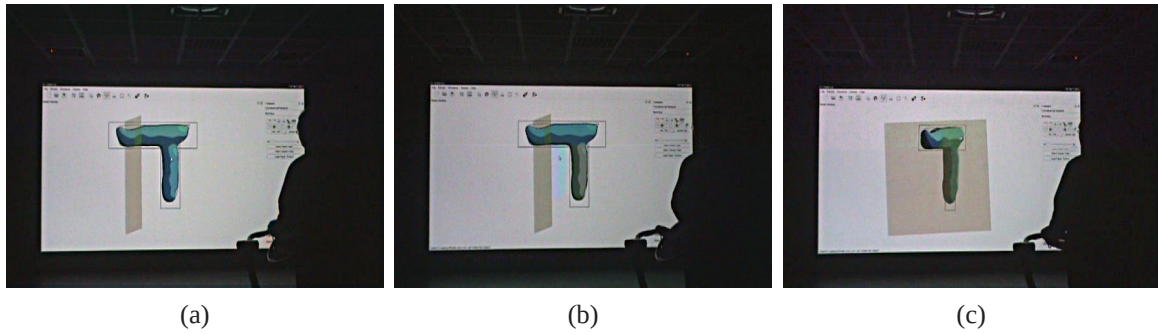


Fig. 1.16: 3D positioning of a new virtual plane. Once the position of the plane is validated by users (a), a translation to the center is applied on both the plane and the object (b). User moves back to horizontal position.

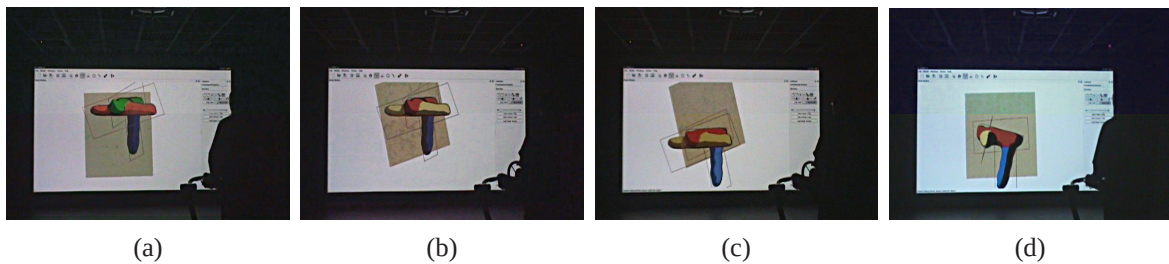


Fig. 1.17: Selection of an object component for edition. From the current position (a), user moves the virtual paper sheet to select another component (b). Each component has its own color (red is reserved for the selected object). Once the selection is accepted by users (c), a translation to the center is applied on both the plane and the object (d). Users move back to horizontal position.

CAT table top has to be vertical. Once back in sketching mode, the rotation of the CAT is applied to the virtual paper sheet and to the objects: users may want to move back the virtual paper sheet to a nearly horizontal configuration, more comfortable for sketching.

This transition is also illustrated³ in Figure 1.16. Once the position of the virtual plane corresponds to users' wish, the transition to sketching mode consists in a translation of the whole scene (object and plane) to the center of the screen. This translation allows users to quickly bring back the virtual sheet in front of him: the only action required is the rotation back of the CAT. During the whole process, the orientation of the CAT table always corresponds to the orientation of the virtual paper sheet. This sequence of actions is similar when it comes to select an existing component (cf. Figure 1.17).

³The images are samples of the associated video. The difference in color between the upper and lower part is due to our stereo system.

1.4 Conclusion

Through this exploration of sketch-based free-form modeling, we have experimented two context-dependent interactions: stylus for small devices or mouse for classical workstations, and special tailored devices for large displays. In each context, we have first investigated what would be the possible more comprehensible actions that would result in the desired shape. Based on these actions, we have thus defined new representations and reconstruction algorithms that have to be involved: we try to preserve comprehensible correspondences between the defined interactions and the resulting representations and reconstruction algorithms. Based on these assumptions, we have first developed modeling tools for meshes using profile curves.

To ease the use of such modeling tools, and to introduce more flexibility in the modeling process, I believe that it is important to expose comprehensibly the different steps of the reconstruction process together with the underlying representation⁴. From the definition of user interactions, we have to first define the underlying representation and the reconstruction process. Once this pipeline defined, we have to expose this representation and this process with tailored user-interactions and visual cues: users are now directly integrated into the reconstruction pipeline. Based on this assumption, we have developed a multi-view interface dedicated to our profile-based sketching, and to 3D positioning interactions for sketching in front of large displays.

Our work leads to more versatile modeling tools for meshes, showing that discrete surfaces are a viable alternative to implicit surfaces. But robustness have to be improved in order to prevent un-desired behavior and failures that might break the interactions between users and modeling process. The main remaining problem in our approach is the transition between different profiles: we have to first define what might be the generally expected result from users, and thus to create the corresponding representation and modeling process. Currently, none of the different proposed solutions [LGS06a, LGS07b, LG07] are fully efficient.

More generally, new possible actions have to be developed in order to increase the richness of the resulting models while preserving the intuitive aspect of sketch-based modeling. The resulting new tools have to be embedded in one unique and intuitive framework. Painting or Shading based [GZ08] approaches are good candidates for increasing the local shape details. Since all the existing tools are currently dependent on the choice of the underlying 3D representation, some work must be done to either convert them to a unified representation, probably based on discrete surfaces such as point-based surfaces or meshes, or to allow the possible on-the-fly conversion between this unified representation and one compatible with the modeling tool.

The results presented here are definitively not a full achievement of our research in this domain, but represent some preliminary validations of our approach for the development of modeling tools. As previously said, we have to work more closely on the cognitive and perception aspects of this research, for both defining new interactions. As an example, a large range of shape might be perceived as similar [Hof98], leading to the following two developments: the reduction of 3D modeling constraints for more freedom in modeling, but also, the accurate visualization of a 3D shape.

On important point is that sketching tools are hybrid solutions: 2D actions for 3D modeling. As stated in Section 1.3, full 3D interactions require either intuitive 3D devices or a very efficient 3D rendering systems. Such rendering systems are still in development and will not reach a full maturity in a middle term period: work has to be done on the accuracy of such systems, based on our perceptions of a 3D space. Moreover, a lot of 3D interactors are based on the use of 2D tablets.

⁴Andrew Woo, who was working at AliasWavefront as the head of the research and development team, has also claimed in one of his talks, that no parameters have to be either hidden, or limited.

In real world, sculpting is often assimilated to 3D modeling. Even though, most of the time, it relies on 2D interactions on the surface to progressively remove or add some components. In conclusion, I believe that sketching and painting in 2D for 3D actions will still remain one of the main user-friendly approaches for 3D modeling.

Local Illumination and Shading

3D modeling is generally organized in three components: geometry, animation, and appearance (i.e., lighting behavior). We have shown in the previous chapter that over the years, and with an increasing interest since the publication of Teddy [IMT99], simple sketch-based interfaces have been proposed for geometry modeling and more recently, for animation (e.g., [TBv04]). This research in intuitive modeling has only recently emerged for the design of lighting behavior or shading (e.g., [OMSI07, TABI07]), but is still in its early stages.

One can first consider shading on the physical side, trying to reproduce the real behavior of a surface toward incident lighting. A lot of research has been done in this domain, from empiric models of reflectance or BRDF (Bidirectional Reflectance Function [NRH⁺77]) to complex acquired materials like Bidirectional Texture Functions (BTF [DvNK97]) or reflectance fields [DHT⁺00]. The main goal of most of these researches was to provide more and more accurate representations for more and more complex and rich materials. In this chapter, we introduce a new approach for modeling BRDF that extend the user freedom in designing lighting behaviors, from plausible to non-realistic ones.

Moreover, the appearance of an object is related to both its lighting property but also, to its geometric characteristics [VLD07]: shading participates to the shape perception and geometry participates to the perception of the lighting property. To improve both shading and shape perception, it is thus important to correctly analyze the shape characteristics. The result of such an analysis would help in providing users with intuitive tools to control the relation between shape features and shading. In this chapter, we introduce a new shape descriptor called Apparent Relief, and its application to a collection of non-photorealistic styles.

The content of this chapter has been validated by the following publications: [PGSP08, VBGS08a, VBGS08b]

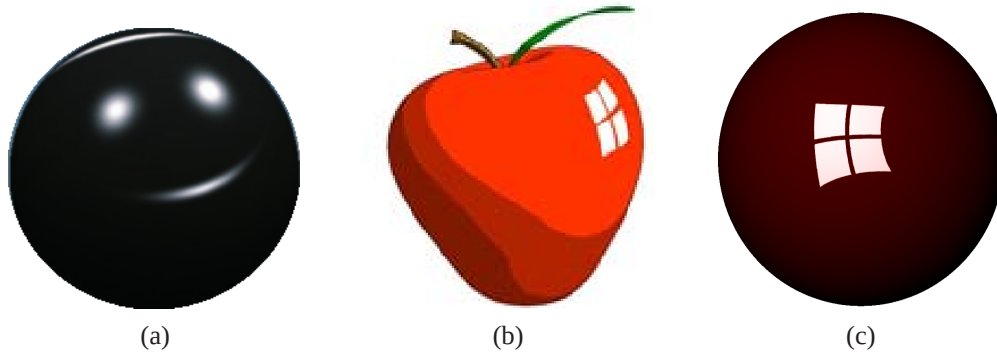


Fig. 2.1: Three different styles of highlight obtained with different systems. (a) BRDF-Shop physic-based highlight, (b) Anjyo [AWB06] cartoon-based highlight, and (c) example of highlight obtained with our system.

2.1 User-Designed Glossy and Specular BRDF

Over all possible lighting effects, some of the most noticeable are specular highlights: they play an important role in the final appearance of an object, providing users with convincing materials. Unfortunately, a highlight is controlled by the lobe of specular or glossy BRDFs, and it is thus a complex 4D function that depends on both light and view directions. Therefore, in a creative and artistic context, edition and creation of plausible BRDF are still challenging tasks. Currently, most systems rely on the selection over pre-defined shading models (such as Phong shading) and the modification of their intrinsic parameters: the users' choice and freedom is thus limited. Recent approaches extend the modeling freedom by using a painting approach: users create and edit BRDFs by painting (cf. Figure 2.1-(a)) the expected result on a sphere [CPK06]. This approach is still limited by the underlying analytical models.

We thus develop an intuitive and flexible system for BRDF design. Our system allows users to specify interactively through sketching, painting, and manipulation of vector data, the lobe's shape and its color gradient. With our new modeling tools, users can create a wide variety of appearances. Thanks to the simple underlying model, our approach can provide real-time feedback and interactive rendering. These results are due to our two main contributions:

- we provide **new sketching and painting tools** associated with gradient edition in order to define easily different BRDF's lobe characteristics, such as their shape and color variations.
- we introduce a **new BRDF model** based on a curve as a global shape representation for its lobe, and a texture for its color and refined shape description.

2.1.1 Previous Work

In Computer Graphics, one of the common ways to represent surface appearance is to use a BRDF (Bidirectional Reflectance Distribution Function). However, its specification relies on a non-intuitive choice by an artist of a pre-defined model (e.g., [Pho75, Bli77, CT82, HTSG91, War92, Sch94a]), and an adjustment of its parameters that can have a non-uniform perceptual behavior. For an improved selection of the expected appearance, Ngan et al. [NDM06] introduced a perceptually uniform navigation in a space defined by the different models and their parameters. On the edition side and once a given model has been selected, Ben-Artzi et al. [BAOR06] proposed to project it into a given basis and to factorize it in a set of 1D curves that can be edited directly, even under complex illumination.

However, even if these solutions provide users with improved selection and edition tools of predefined BRDFs, they do not allow the creation of more freely designed ones.

To increase modeling freedom, Colbert et al. [CPK06] develop Poulin and Fournier’s work [PF95] by proposing a painting interface. In their tool, called *BRDF-Shop*, different “painting” operations are introduced for the design of highlights. The resulting BRDF is approximated by a sum of Ward lobes [War92], but other lobe models can be used for this fitting process [NDM05]. Their approach does not guarantee that the painted highlights correspond to a reasonable fit because an arbitrary choice on the underlying analytical model has been done, a non-linear fitting does not guarantee to reach the best approximation, and a large number of lobes can be required. An improved fitting process or an increasing number of lobes will thus result in reduced interactivity. In contrast, Edwards et al. [EBJ⁺06], extending the work of Ashikhmin et al. [APS00], create a unique lobe by designing a probability distribution function of normals. A similar indirect control has been proposed also by Neumann et al. [NNSK99], where the shape of a lobe is defined on the tangent plane of the surface. Generally, the main control on BRDF is done indirectly by designing a micro-geometry [WAT92, Sta99, APS00] or a similar bump-map [CMS87]. Unfortunately, these indirect controls can be non-intuitive.

By removing constraints on realism, more freedom is provided to users in highlight design in a non-photorealistic context. One of the first solutions, based also on a painting metaphor, is the *Lit Sphere* [SMGG01] by Sloan et al.. In their approach, the painted appearance on a sphere, for given viewpoint and light direction, is used as a texture projected on 3D surface, taking into account the similarity of the configuration between the viewpoint and the normal. Therefore their approach is only possible for a fixed lighting direction. Recently, Okabe et al. [OMSI07] directly painted the lighting on a 3D surface. From this input, a 3D environment map is constructed and used as light source, but no control on BRDFs can be provided. Anjyo et al. [AWB06] tweaked the shape of a highlight in cartoon-like (cf. Figure 2.1b) shading. However they do not really edit the BRDF: they locally move, scale, split, and merge the light sources in order to obtain the requested shape which is the only editable parameter. Smoothness, glossiness, and color variations are not taken into account. Similarly, Todo et al. [TABIO7] remove or add some highlights in toon shading, using offset texture defined for light key-directions. Unfortunately, this solution is limited to stylized shading.

Of all the previous solutions, the most closely related to sketching is the work of Pellacini and Lawrence [PL07]: they use strokes to select different areas of a BTF to interactively modify materials. In our approach, we also want to provide users with a fully interactive edition, but for the design of BRDFs. For user friendliness, our solution is based on sketching and painting metaphors. For efficiency and interactivity reasons, we directly manipulate the shape and colors of BRDF’s lobe.

Overview

As illustrated in Figure 2.2, our approach is based on the direct edition of features of BRDF’s lobe, displayed on a plane oriented perpendicularly to the light mirror direction $\mathbf{r}$. On this plane, the artist changes the shape of BRDF’s lobe and color-gradient with 2D editing tools. Consequently, the lobe features are defined for a given light **key-direction** $\mathbf{l}$, similar to the solution of Todo et al. [TABIO7]. The set of lobe features associated with a light key-direction is called a **lighting configuration**. A typical workflow in our system is firstly to define the lobe characteristics with our 2D tools (first shape, then color and gradient, as shown in Figure 2.4), and secondly to observe the behavior of resulting highlight for different light or view directions. By default, our system replicates the same lobe features for all lighting configurations. However, users can edit them for a defined light direction. For undefined ones, our system smoothly interpolates every lobe features to ensure a coherent behavior.

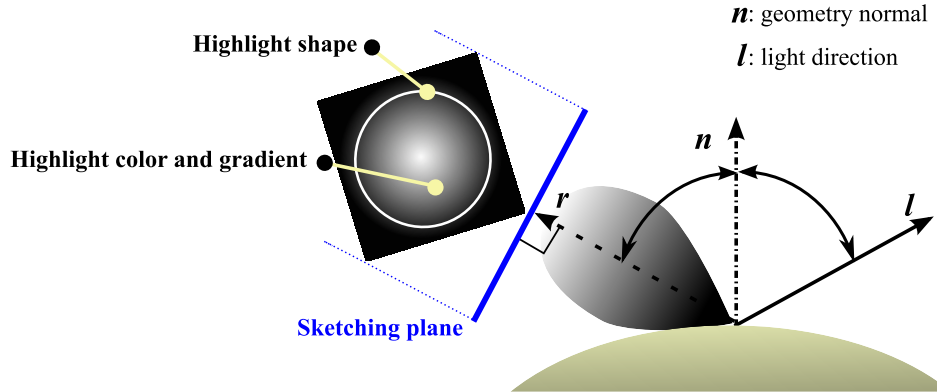


Fig. 2.2: Sketching plane. Users sketch the shape of BRDF's lobe and paint its color on a Sketching Plane oriented perpendicularly to the light mirror direction r .

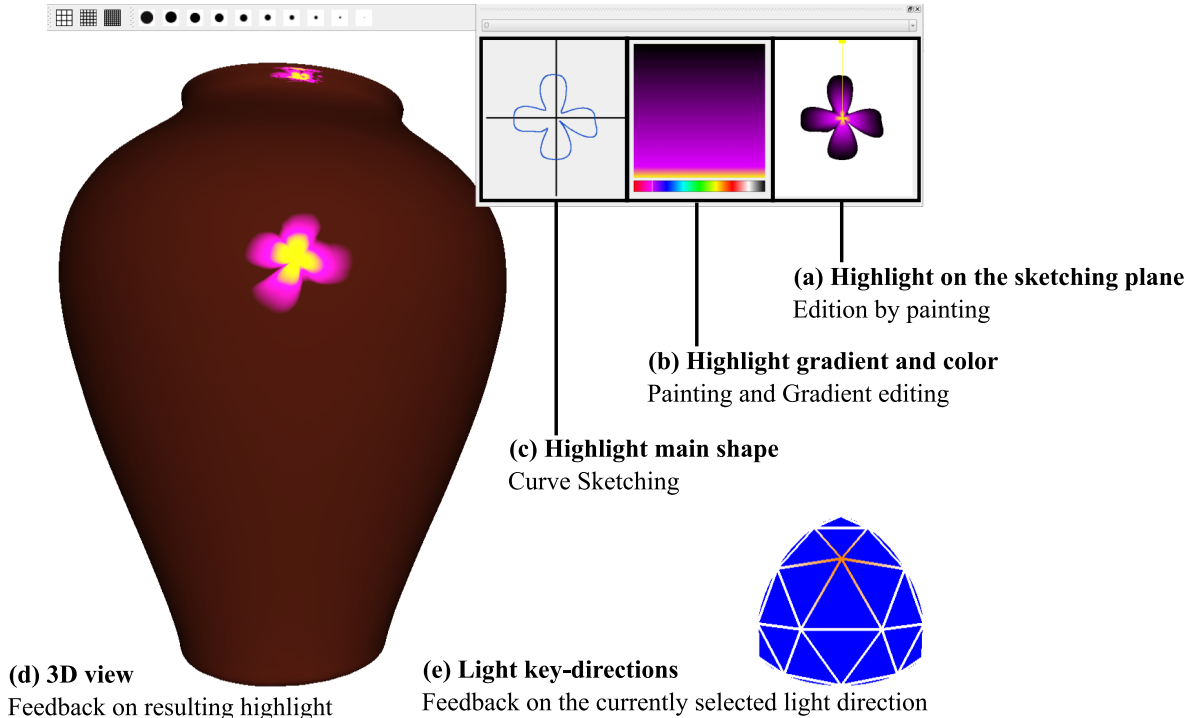


Fig. 2.3: User interface main view.

Furthermore, at any time our system provides real-time feedback to users' editing actions.

2.1.2 User-Controls

To provide intuitive tools for BRDF modeling, we rely on three different interaction approaches adapted for each modeling action. For simplicity of use, each interaction is associated with one specific screen area (cf. Figure 2.3). As illustrated in Figure 2.4, users can sketch the shape of BRDF's lobe, paint its color or edit its gradient with vector tools.

Sketching has been recognized as an efficient tool to define a global shape. Therefore, we rely on this approach to modify the shape of BRDF's lobe. Through sketched strokes in a specialized area (cf. Figure 2.3-(c)), users select part of or the whole lobe shape (cf. Figure 2.4-(a)). Once selected,

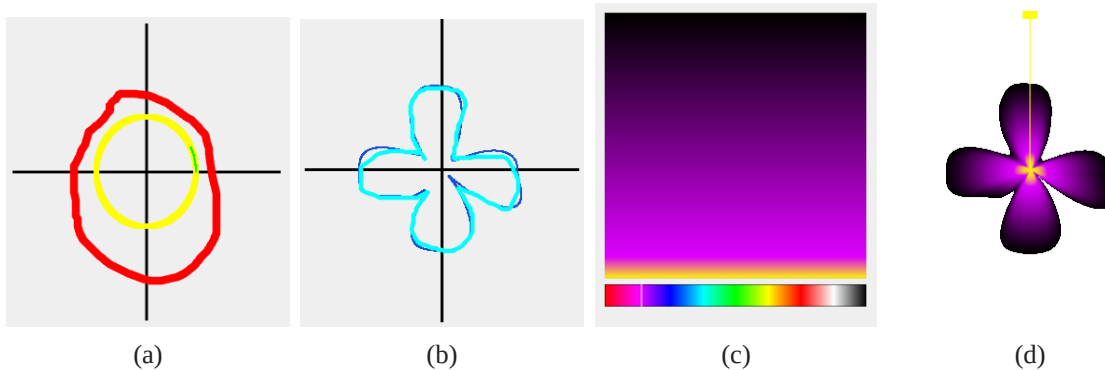


Fig. 2.4: (a) The external curve represents users' selection sketch. (b) The cyan curve represents users' sketch whereas the blue one represents the fitting result. (c) A user-defined color gradient. (d) Sketching Plane view that displays the resulting highlight when applying (b) and (c).

a new target curve is drawn. A fitting process then approximates the sketch (cf. Figure 2.4-(b)) using the underlying representation of curves (defined in Section 2.1.3). This sketch-based definition of the shape also controls the highlight shininess: the smaller the shape is, the shinier the highlight is. To increase the shininess, users have simply to scale down the curve.

To define color variations and gradients, painting is the most suited approach. Users thus use a set of brush and gradient tools to refine the lobe shape and to control its color behavior. Brushes allow users to edit the color while the filters let her adjust the intensity. This is also done in specialized areas, one for the gradient manipulation (cf. Figures 2.3-(b) and 2.4-(c)), one for a precise edition on the final results (cf. Figures 2.3-(a) and 2.4-(d)). The combination of the shape and the gradient texture is explained in Section 2.1.3. If the embedded tools are not sufficient, an experimented artist can load an image created with any other image manipulation software.

For an interactive modeling system, real-time feedback is crucial. Thanks to our representation based on a unique lobe defined with a simple curve and a texture (see Section 2.1.3); each modification is directly viewed on a selected 3D model (cf. Figure 2.3-(d)). Since the highlight depends both on the light and on the view directions we provide also users a visual representation of the current light key-direction (cf. Figure 2.3-(e)). Furthermore, users can select a part of the 3D object to retrieve local parameters, such as the most important light configuration or the highlight color and shape. Although this is not a 3D painting interface, it greatly facilitates the users' work when it comes to modifying precisely parts of the highlight.

2.1.3 BRDF Model

For each lighting configuration, a distance field defined by a curve represents the lobe shape and a texture represents its colors and intensity.

Lobe Shape

We define the lobe shape as a curve, the size of which controls the highlight **shininess**. The key idea here is to use this curve as the outline for the whole lobe. Outside of this curve, the reflected intensity is null, whereas inside the intensity is modulated by a color texture. In order to fill the shape, our representation should allow a simple and intuitive access to the color texture. Furthermore, we also need a representation that can be easily interpolated between light key-directions. This is required for the reconstruction of lobe shape for unspecified light directions.

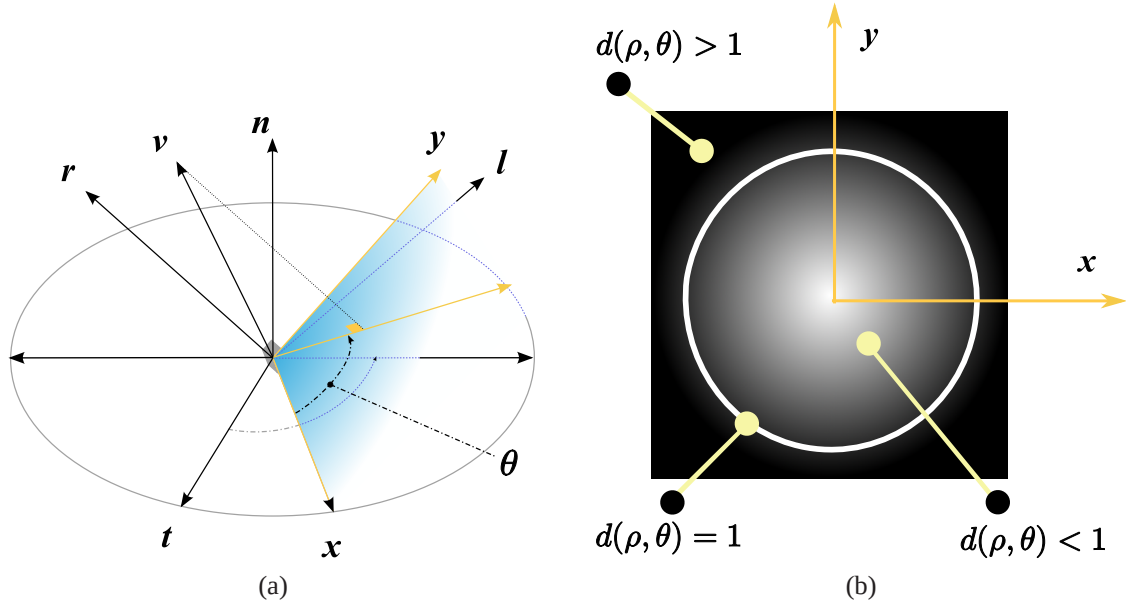


Fig. 2.5: (a) The x and y directions constitute the Sketching Plane local frame. They are computed according to the light mirror direction r and the geometry local tangent t . To compute the reflected intensity, we project the view direction v on the Sketching Plane. (b) The distance field $d(\rho, \theta)$ defined by the polar curve parameterizes the Sketching Plane.

Therefore, we use the spline-based polar curve proposed by Crespín et al. [CBS96]. Each control point for the curve is located by an angle θ and is defined by its radius ρ , and its left and right tangents. These parameters (radius and tangents) are easily interpolated. Furthermore, this curve representation enables the definition of a distance field $d(\rho, \theta)$ that starts from the center of the Sketching Plane (cf. Figure 2.5-(b)). This distance field is then used to fill the shape with a color texture.

Texture Definition

The color texture represents the reflected color and intensity inside the shape. Therefore, black texels indicate areas where the reflected intensity is null. This allows a refinement of the curve-defined shape. Furthermore, by controlling the intensity gradient, we control the highlight **glossiness**.

Two parameterizations (polar or Cartesian) are used for texture lookup. We found it easier to work with polar coordinates when the texture is used to define a simple color gradient, and Cartesian coordinates for more generic textures, such as those created using painting tools. When using the polar parameterization (cf. Figure 2.4), the horizontal axis of the texture color component represents the angular variation whereas the vertical axis represents the radial one.

Since the color texture is enclosed by the star-shape polar curve one, we can also use it to define more complex shapes. To further ease this process, we separate, as it is done in common painting software, the color texture into two multiplicative layers. One layer stores the reflection color and the other stores the reflection intensity as gray level (cf. Figures 2.9 and 2.10).

Parameterization

Finally, we need to define the parameters ρ and θ as functions of the view and light directions:

$$\begin{cases} \rho = {}^T\mathbf{v}\mathbf{r} \\ \cos \theta = {}^T\mathbf{v}\mathbf{x} \\ \sin \theta = {}^T\mathbf{v}\mathbf{y} \end{cases} \quad (2.1)$$

where T denotes the transpose operator, $\mathbf{r}$ the light mirror direction, and $\mathbf{v}$ the view direction. The directions $\mathbf{x}$ and $\mathbf{y}$ are defined as follows:

$$\begin{cases} \mathbf{x} = \mathbf{t} \times \mathbf{r} \\ \mathbf{y} = \mathbf{r} \times \mathbf{x} \end{cases} \quad (2.2)$$

where $\times$ denotes the cross product, and $\mathbf{t}$ the local geometry tangent (cf. Figure 2.5-(a)). With polar coordinates, to access the color texture, the system uses the following (u, v) texture coordinates:

$$\begin{cases} u = \frac{\rho}{1 - d(\rho, \theta)} \\ v = \frac{\theta}{2\pi} \end{cases} \quad (2.3)$$

To summarize, for a given light direction, the system smoothly interpolates a distance field $d(\rho, \theta)$ and a color texture that are both used and accessed, at rendering time, according to the current view direction.

Implementation

We only use the CPU to perform the least-square fitting of the polar curve. To achieve interactive feedback, we rely on current graphics hardware capabilities using *OpenGL Shading Language*.

The shape of the BRDF's lobe is stored as a floating 3D texture. A slice of the 3D texture contains parameters (radius and tangents) of one curve's control point for each lighting configuration. The number of slices is the number of control points of the polar curve. Therefore, a slice parameterizes a spherical triangle representing all possible lighting configurations. We pack every color texture (one for each lighting configuration) into a single texture array (ARB_texture_array extension).

In practice we use between 20 and 60 control points with 3 different lighting configurations, thus requiring less than 40 KB to store all the features (shape + color). Therefore, our representation is a very low-consumption memory solution and one may use it to store many different shaders on GPU resident memory.

2.1.4 Results

We perform all the experiments on an Intel Core 2 Duo T7500 CPU workstation with a GeForce 8600 M GT. For a rendering resolution of 1280×1024 and an object complexity of 35 000 triangles, we report a frame-rate of 60 frames per second. On the interaction side, each action leads to an update of the interface in less than 100 milliseconds. Through all the results, keep in mind that the final appearance results from the combination of diffuse and highlight colors. For these results, we focus on illustrating the large range of possible glossy BRDFs, with quite complex and often non plausible

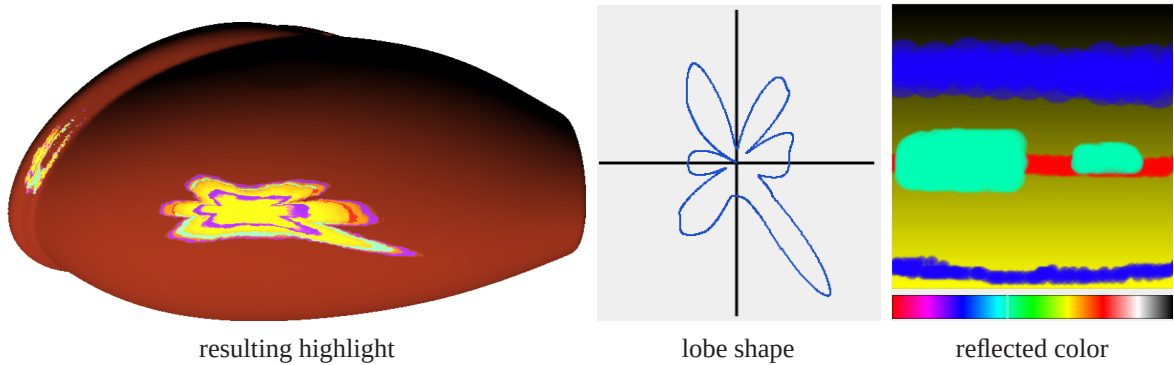


Fig. 2.6: A more complex model. Note the color variation and the highlight shape deformed by the curved geometry.

shapes. Note that, as illustrated in Figures 2.7 and 2.8, we can also create more easily plausible BRDFs, using simple shapes and realistic color of its lobe.

Figure 2.7 shows a scenario where users sketch a trefoil-like shape with different sizes and gradients for each light key-direction (cf. Figure 2.7(d)). For the normal incidence direction, the color texture is a green gradient; at tangent (resp. bitangent) incidence, the color texture is a blue (resp. red) gradient. As demonstrated in Figures 2.7(b) and (c), when the light shifts toward the tangent (resp. bitangent), the highlight becomes more blue (resp. red). The yellow color in Figure 2.7(c) comes when interpolating between the red and green colors stored for two lighting key-directions. This explains why the upper part of the sphere, where light configurations for normal and bitangent key-directions have equal importance, is yellower while the lower part of it, where the configuration for bitangent direction is more important, appears redder. Users can sketch more complex shapes, such as the one presented in Figure 2.6. Contrary to the previous example, painting tools were used to create a more complex color texture. For both previous examples, our system uses polar coordinates for texture look-up.

With a similar approach, users can create complex appearances, close to realistic ones, such as the dispersion-like effect of Figure 2.8. Our previous model, based on approximations of physical behaviors [GH03], provides users with complex and un-intuitive controls. Based on his experience of *Diffraction Shader* [Sta99], Jos Stam has also introduced a simplified version for a GPU implementation [Sta04]. Even simpler, the physically-based parameters reduce user freedom. With our new representation and user controls, a similar effect is simply done by creating two different color textures (using Cartesian coordinates) and two different shapes (cf. Figure 2.8(d)). For the lighting configuration at normal incidence, the texture is a white gradient, where for grazing configurations, it is a rectangular rainbow gradient. Furthermore, a circular shape is used for the normal incident light key-direction and a rectangular shape for the others.

The two layers of the color texture can be used to refine the star shape created with the sketched curve (cf. Figures 2.9 and 2.10). We model the BRDF's lobe of Figure 2.9 with a smiley bitmap texture (cf. upper right corner of Figure 2.10). As explained earlier, points of the sketching plane falling outside the bitmap texture are not shaded (i.e., the reflected intensity is null). Since the color texture is enclosed in the polar curve shape, it can be deformed to achieve new effects, like the unhappy smiley of Figure 2.9-(b). Finally, Figure 2.10 shows an example with more complex color texture. The color layer is filled according to the lighting configuration, whereas the intensity layer, which contains the shape, remains constant (cf. Figure 2.10-(d)).

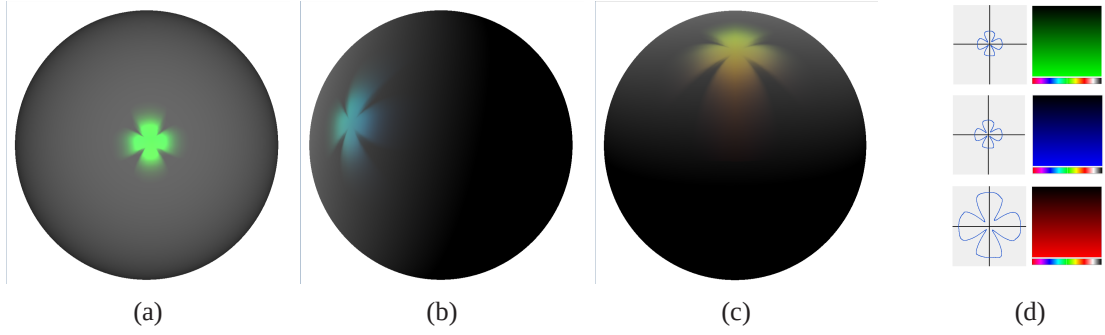


Fig. 2.7: Light key-directions illustration. The highlight color and size is set according to the light direction. When the light direction is collinear with the geometry normal (a), the highlight exhibits a green color, whereas when the light shifts toward the tangent (b) (resp. bitangent (c)), the highlight becomes more blue (resp. red). For configurations (b) and (c), the highlight size is larger than for (a) to increase the color-shift effect.

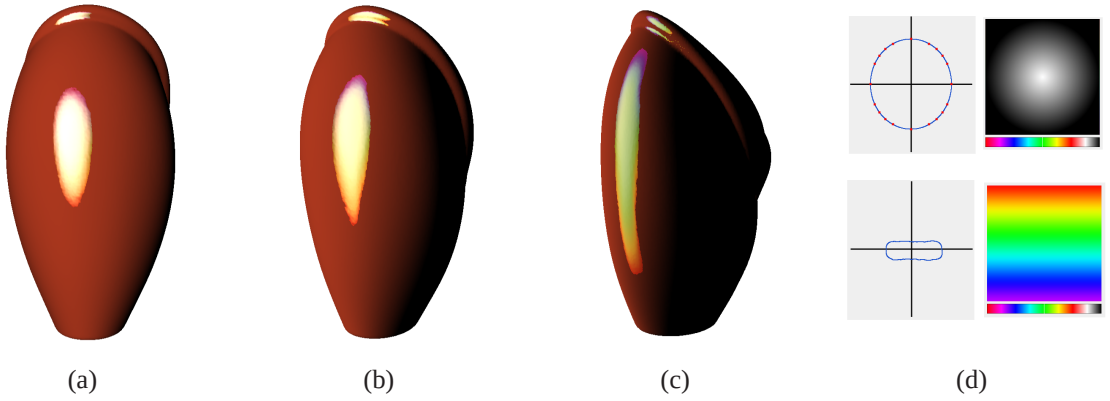


Fig. 2.8: Dispersion example. (a)-(c) The highlight exhibits a dispersion behavior according to the light direction. Top (resp. bottom) row of (d) shows the color and shape configuration for the configuration at normal (resp. grazing) incidence..

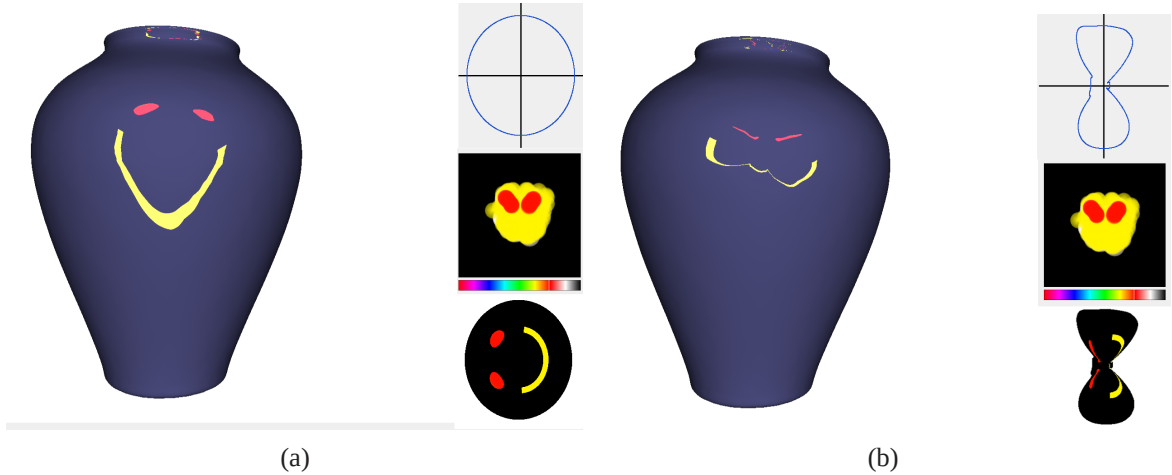


Fig. 2.9: Bitmap deformation example. (a) The initial shape of BRDF's lobe, generated with a smiley-like bitmap texture, is changed to (b) an unhappy smiley by modifying the shape curve.

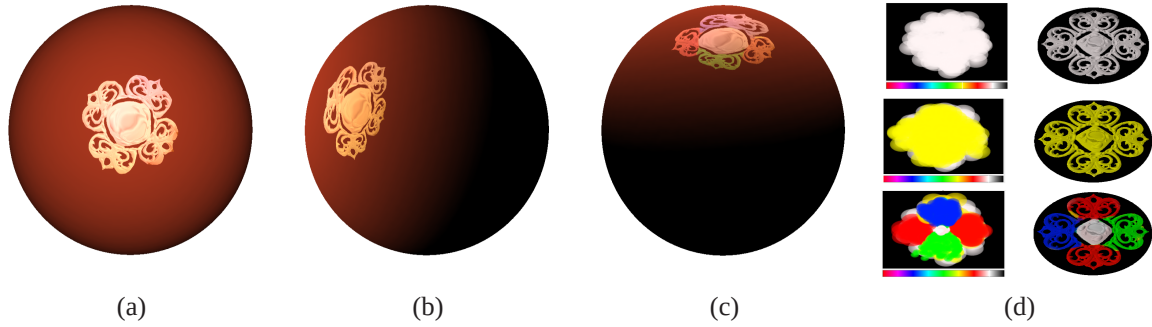


Fig. 2.10: (a)-(c) Highlight rendered under different light directions. (d) The first (resp. second and third) row shows the lighting configuration when the light is collinear with the geometry normal (resp. tangent and bitangent). All configurations use a circular shape.

2.1.5 Limitations and Possible Extensions

Our system and its BRDF representation are only a first step and some limitations remain. First, since the painted and sketched features are projected from the sketching plane on the surface, which is not necessarily planar, some shape deformation can occur. However, our system allows changes and adjustments of the shape in real time, such that users may compensate for this deformation. This was done, for example, in Figures 2.9(a) and (b) to compensate the vase curvature. Even though our curve-based representation ensures a smooth interpolation between different shapes, some interpolation artefacts may occur when using complex color textures (as in Figure 2.10). This limitation is related to the unsolved general problem on how to morph smoothly between two arbitrarily different images. Third, remember that our system models a BRDF's lobe expressed as a function of the view and light directions. However, these directions are parameterized according to the local geometry normal. Therefore, a complex surface with many normal variations exhibits many highlights but not necessarily on a large area. This explains why highlights in Figures 2.8 and 2.9 on the upper edge of the vase are almost indistinguishable.

Although it might still be complex to design a BRDF from 2D sketches and paintings rather than directly paint it on the object, we think that this drawback is compensated by our interactive feedback and our visual hints. Finally, even though we show examples with only three different lighting configurations, one may use more lighting configurations (within hardware limits). However, we found, through empirical testing, that increasing the number of lighting configurations increases the BRDF design complexity for the artist.

Future Work

We have introduced a new system based on sketching and painting metaphors for BRDF design. A lobe of our BRDF is represented with a polar curve for its shape and a texture for its color gradient and for precise shape enhancement. Our simple and compact representation, suitable for hardware rendering, allows real-time edition of BRDFs, which can be various and span from realistic to non-photorealistic appearance. Our tools and representation also provide more editing and creative freedom than previous approaches. I believe that our solution is more intuitive for BRDF control in highlight design.

Our upcoming work follows two streams. First I would like to extend it as a full 3D painting and sketching interface where users can draw directly the appearance for simple and complex illuminations. I believe that this would be possible with the same parameterization, but a new representation,

like a pure image-based representation using higher-dimensional texture or using multiple 2D textures as in the homomorphic factorization [KM99]. In the restricted context on realistic or plausible BRDF, I also believe that such an image-based representation is well-suited: by initializing the representation with a user-chosen model, and by converting in minimal changes that fit user-painted modifications, the original plausible behavior might be preserved. In general, further investigations have to be done to define new BRDFs models that can accurately represent physical behaviors while still providing users the possibility to extend the range of possible lighting to more creative ones.

Second, new tools and a new highlight model with spatially varying properties have to be developed to help users to control highlights on highly detailed geometry. In the next section, we investigate a first approach to provide users with a detailed description of the geometric details in order to either preserve them into different shadings or, more important in the current context, provide users with a possibility to select shading model according to selected geometric features.

2.2 Shape Depiction through Expressive Shading

As seen in the previous Section, the final appearance of an object does not only depend on the definition of its shading property, and also on the underlying geometry. As seen in Figures 2.6 and 2.8, multiple highlights occurs on the top of the vase due to denser geometric details than for the rest of the vase. Moreover, our perception of reflection properties also highly depend on the local shape features [VLD07]. For a user's point of view, it would be convenient to be able to locally adjust the shading parameters according to the local shape characteristics. On the other side, these local modifications on shading can also help in improving or reducing local shape depiction, since shape information is perceptually conveyed by its shading.

Shape depiction is an important dimension of image creation for a variety of reasons. In scientific illustration, shape depiction techniques are used to remove possible ambiguities in visual interpretation. They are often seen as processes that highlight the most salient characteristics of an object's shape. This is often done at the expense of other details which are removed because they are irrelevant to the information the image is supposed to convey [Woo94]. In paintings and drawings, shape depiction techniques may be used in more subtle ways, and for other purposes. For example, they might help understand relationships between characters and background; clarify a representation that makes use of drastically simple shading rules; or even portray hidden portions of surfaces like in cubist paintings.

Providing an intuitive and efficient control over the depiction of 3D objects' shape in Computer Graphics is thus of primary importance. Many previous approaches focus on a single body of techniques: line-based rendering. The goal of such techniques is to generate some *shape cues* to depict the essential characteristics of an object that correspond to *shape discontinuities*. For instance, contour lines may represent discontinuities of depth, curvature, orientation, object IDs, etc... Line-based techniques have been used in a variety of applications, from architecture to video games. They represent only a subset of shape depiction techniques though. In particular, many artists rather depict an object's shape through shading: as an example they may use gradients, which are other kinds of shape cues that correspond to *continuous* variations on the shape. Some reasons for the choice of such a style are that it allows the artist to convey more subtle information about shape, and that it is integrated seamlessly into conventional lighting [Hog91].

Previous methods often imitated traditional illustrations to depict shape through shading (e.g., by creating artificial light sources that indirectly convey shape information); unfortunately, such methods quickly become cumbersome to manipulate in practice. For this reason, we rather take the approach of directly extracting and manipulating information issued from shape analysis. The main issue is that this information can no longer be defined by sharp *discontinuities* as in line-based rendering, because when integrated into shading, they would have little influence. Instead, we seek a set of *continuous* values that have to be defined for each pixel of an image. The goal of expressing them directly from the 3D data that defines object's shape would be rather complex. Therefore, an intermediate representation, that we call a shape descriptor, is needed to assist users.

We thus introduce a view-dependent and continuous shape descriptor called Apparent Relief, from which shape features are easily extracted, and which gives rise to stylized shading-based shape depictions. By construction, it naturally leads to *automatic* Levels-of-Detail effects: when the object gets closer or farther from the point of view, finer or coarser shape features are revealed respectively. This approach runs in real-time on modern graphics hardware, and is also simple to implement. We illustrate its potential using four shading styles: cel shading, cartoon shading, minimal shading, and exaggerated shading.

2.2.1 Previous Work

Many previous approaches focus on line-based renderings to depict shape in a non-photorealistic way. A conventional method consists in analyzing differential geometry to identify maxima and minima of curvature (i.e., third-order discontinuities), such as ridges and valleys [OBS04]. Other techniques take the point of view into account to extract either silhouettes [Her99] or suggestive contours [DFRS03, DFR04, DR07], which extend and anticipate silhouettes. A line-based rendering can also be obtained from pure image-space attributes. As examples of attributes are discontinuities of depth, object IDs [ST90, Her99], or directions of normal vectors [Dec96, ND05].

With object-space techniques, contour lines are explicitly extracted as vector primitives, hence opening their rendering to stylization. Their stylization is not a trivial process though, as *temporally coherent* stylization is a complex task [KDMF03]. Moreover, the number and location of lines need to be adapted based on the viewing distance [BTS05], a process similar to the creation of Levels of Detail (LODs). Some solutions work by using geometry simplification as a pre-process [JNLM05, NJLM05], but there is currently no method to deal with this issue dynamically. Image-based methods avoid the need for geometry simplification, and are thus naturally adapted to LODs: as the number and location of extracted lines directly depend on the projected size of the 3D object, LODs are managed automatically. On the other hand, since lines are only identified as pixels, they do not have direct access to object-space attributes such as surface convexity, and are not easily stylized. Our approach combines advantages of object- and image-space techniques.

Recently, a new method called *Apparent Ridges* [JDA07] has been introduced to provide a generalization to previous line-based rendering techniques. It relies on the idea that apparent object-space contour lines should be extracted by tracking the discontinuities of a view-dependent shape descriptor. In the case of line-based shape depiction, apparent ridges provide quite convincing results. However, they cannot be directly employed to fulfill our goal of shading-based shape depiction: they do not provide continuous shape information, are ill-defined at silhouettes and require further specific LOD mechanisms depending on the viewing distance. This work inspired ours though, and we give comparisons with Apparent Ridges in Section 2.2.1.

Another approach to shape depiction is to rely on a notion of *accessibility*. The original accessibility shading method [Mil94] relies on a geometric shape descriptor: accessibility is defined as the maximum radius of a sphere touching the point of interest without intersecting other portions of the surface. Another descriptor has gained a lot of interest with the so-called ambient occlusion technique [ZIK98]: in this case, the portion of visible hemisphere characterizes its accessibility. Unfortunately, the resulting shape cues only consist in scalar values at each point which brings only information of deep concavities. Moreover, both accessibility methods usually demand large precomputation times, and the interactive solutions for ambient occlusion are not robust [Mit07].

Many NPR shading techniques have also been introduced in previous work (e.g., [GSG⁺99, BTM06]). However, none of them proposed to depict object shape explicitly (i.e., users have no direct control over shape cues). Moreover, they are often restricted to specific styles. Other approaches rather modify lighting [LH06, RBD06], as is the case of exaggerated shading: the idea is to locally adjust the light direction over different areas of a surface, revealing details usually only seen in places where light reaches a grazing angle. The method does not explicitly extract a shape descriptor though, so that the control on users' side is restricted to the exaggeration of any detail on the surface, and stylization is limited to tone variations. In contrast, our approach allows to directly manipulate shape information through a view-dependent shape descriptor, which in turn allows us to control exaggerated shading more intuitively.

Overview: Shape Descriptors

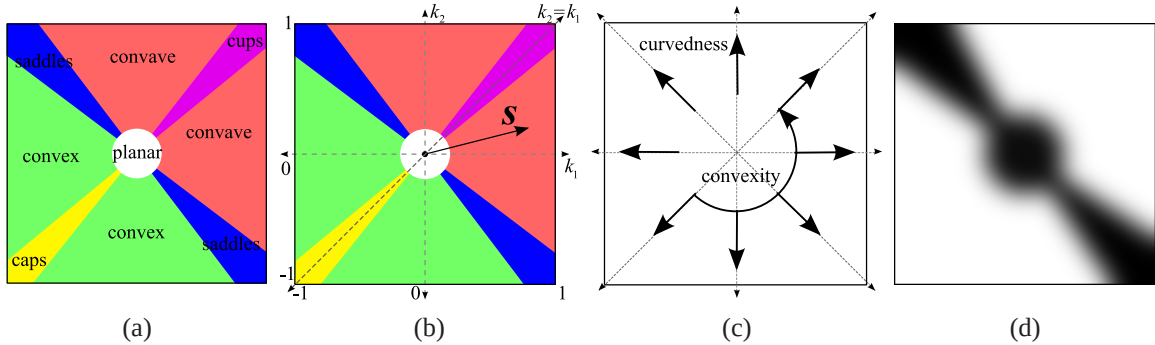


Fig. 2.11: Shape descriptor domain: In (a) we show the color code used throughout the figures of the paper to visualize our shape descriptor: planes are in white, caps (in yellow) and cups (in pink) are located around the symmetry axis, and saddles (in blue) separate convex (in green) from concave regions (in red). As shown in (b), the axes of the parameter space correspond to principal curvatures k_1 and k_2 , where our shape descriptor $\mathbf{S}$ corresponds to a vector. Its length represents the surface curvedness and its direction its convexity as shown in (c). In (d), we show an example of Apparent Relief Map (ARM). Using this texture, shape features are easily selected: here, surface points corresponding to planar- and saddle-like regions are filtered out, as explained in Section 2.2.5.

Our solution to shading-based shape depiction is to manipulate continuous information through a shape descriptor inspired from the work of Koenderink and van Doorn [Kv92]: for each pixel, we extract a descriptor $\mathbf{S}$, which corresponds to a vector in the shape descriptor domain of Figure 2.11-(a). Its direction gives information about surface convexity (cf. Figure 2.11-(b) and (c)), while its length gives information about surface curvedness (cf. Figure 2.11-(c)). Users are then able to select specific shape features via a dedicated texture as the one shown in Figure 2.11-(d): this texture map a continuous and scalar weight for each characteristic value of the descriptor. In Figure 2.11-(d), planar and saddle-like regions of the surface are filtered out.

However, our *Apparent Relief Descriptor* differs from the one defined in [Kv92] as it makes use of two sets of shape attributes: *First*, convexity information is directly computed in object space using differential geometry; *Second*, curvedness information is computed in image-space, incorporating view-dependency and LOD functionalities. The entire process is depicted in Figure 2.12.

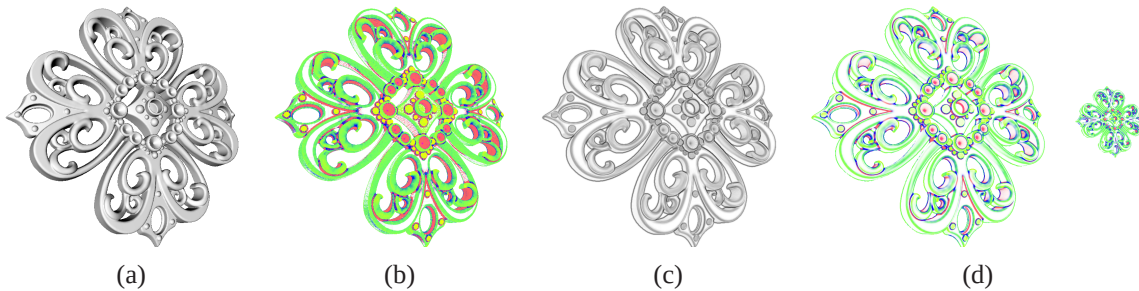
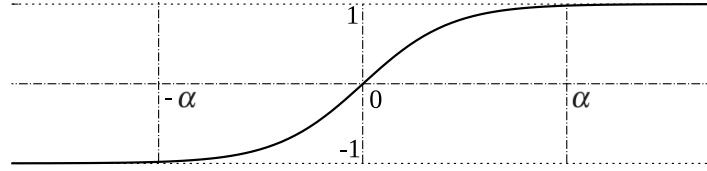


Fig. 2.12: Extracting the Apparent Relief Descriptor: Given an input 3D object (a), we first analyze object-space shape attributes to extract convexity information (b). Then we extract curvedness information from normal variations in image-space (c) and combine both sources in a single shape descriptor: the Apparent Relief (d) - see Figure 2.11-(a) for color code. Note the automatic LODs obtained thanks to image-space measurements in the rightmost image.

2.2.2 Object-Space Information

The shape descriptor of Koenderink and van Doorn [Kv92] describes an arbitrary 3D surface at any point. The intuition behind it is given in Figure 2.11-(a): the axes correspond to principal curvatures k_1 and k_2 , so that any surface point can be assigned a shape vector $\mathbf{S}$ in such a shape descriptor domain. Note that k_1 and k_2 are considered to be in the unit range $[-1, 1]$, which requires some normalization step, detailed below. The shape vector is conveniently defined by its direction $\mathbf{D}$ which corresponds to convexity information, and its length L , which corresponds to curvedness information. Given $\mathbf{K} = (k_1, k_2)$, they are given by $\mathbf{D} = \mathbf{K}/|\mathbf{K}|$ and $L = |\mathbf{K}|/2$. Note the symmetry around the first diagonal in Figure 2.11: this is due to the arbitrary assignment of k_1 and k_2 (the axis of symmetry corresponding to $k_1 = k_2$). Note also that $\mathbf{D}$ is undefined for $|\mathbf{K}| = 0$. It corresponds to the center of the shape descriptor domain, and only occurs in locally planar regions on the surface (we will come back to this singularity in following sections).

The most straightforward approach is to compute the shape descriptor at each point of the surface. However, we must first remap each of the principal curvatures k_1 and k_2 to the unit range $[-1, 1]$. To this end, we use a hyperbolic tangent as a scaling function



$$\mathcal{S}_\alpha(x) = \tanh\left(x \frac{K}{\alpha}\right), \text{ with } K = \tanh^{-1}\left(\frac{2^B - 2}{2^B - 1}\right).$$

In this function, B controls the precision in number of bits (we use $B = 8$). This function remaps the curvedness range $[-\alpha, \alpha]$ to $]-1, 1[$, since for each $|x| > \alpha$, $\mathcal{S}_\alpha(x)$ will be rounded to 1.

There is an important limitation to this straightforward use of $\mathbf{S}$ though: it does not consider view-dependency at all, for instance leaving out most of the information around silhouettes. Note that the view-dependent curvature operator introduced in [JDA07] cannot be used to deal with this issue, as the corresponding principal curvature directions are not orthogonal, a requirement for the use of Koenderink’s shape descriptor.

Our solution to this problem of view-dependency is to define a hybrid shape descriptor combining object-space and image-space attributes. More precisely, the main idea is to directly compute the shape vector’s direction $\mathbf{D}$ in object-space, while deferring the computation of its length L to image-space. This approach is somehow similar to the method used in [JDA07], as they also use pure object-space attributes to discriminate between ridges and valleys in the very end of their pipeline. Our approach is quite different though, as we compute a richer convexity information (not only a binary “ridge or valley” flag), and for every visible surface point (not only onto discontinuities).

In practice, we compute a curvature tensor for each vertex of a triangle mesh using the algorithm in [Rus04]. To ensure that curvatures are continuously interpolated per pixel on the GPU, we sort curvatures according to $k_1 \geq k_2$. Thanks to the symmetry axis, only the lower-right triangular area of the shape descriptor domain has to be considered. However, note that in general, k_1 and k_2 do not respectively correspond to maximum and minimum curvatures.

At this stage, $\mathbf{D}$ is computed independently for every pixel in the picture, so that sharp transitions of $\mathbf{D}$ values may happen in image-space (e.g., between convex and concave regions). It occurs most notably around silhouettes. We wish to avoid such discontinuities, and provide a smooth transition between regions of different convexity information: for instance, we wish to create a saddle at the

T-junction between a ridge and a valley. This can be easily done by integrating principal curvatures in small neighborhoods in image-space: using a smooth integration kernel, we create transitions which vary smoothly in the shape descriptor domain. To this end, for a given pixel (x, y) in the image-plane, we apply a Gaussian blur to $k_1(x, y)$ and $k_2(x, y)$:

$$\begin{aligned} k_1^s(x, y) &= k_1(x, y) \otimes g(x, y, \sigma) \\ k_2^s(x, y) &= k_2(x, y) \otimes g(x, y, \sigma) \end{aligned}$$

where σ is a scale parameter that controls the amount of blurring. Thanks to the separability of the Gaussian kernel ($g(x, y, \sigma) = g(x, \sigma)g(y, \sigma)$), the whole process can be efficiently performed on modern graphics hardware using two rendering passes in pixel shaders. Given a pair of smooth curvatures k_1^s and k_2^s , we compute $\mathbf{D}$, and then proceed to the extraction of L in image-space.

2.2.3 Image-Space Information

We now need to determine the view-dependent curvedness information of our shape descriptor. Another goal is to provide automatic LODs, so that when the object gets closer or farther from the point of view, the shape descriptor will reveal finer or coarser shape details respectively. Our approach is to compute curvedness as the amount of variation across normals in a pixel neighborhood. An important observation is that the variation of normals cannot be accurately computed by only using a scalar function. For instance, one could imagine computing derivatives of surface *slant* for this purpose, defined as the dot product between the normal and view vector. Unfortunately, this will miss many salient shape features such as the front facing edge of a cube for instance (see Figure 2.13 for a more complex example).

The intuition behind our solution is thus to compute curvedness information *both* along the $\mathbf{x}$ and $\mathbf{y}$ axis of the picture plane, and combine them to get L , so that the most prominent shape features are identified in the end. More precisely, we are interested in the variation in normals along each of the picture plane dimensions that is, derivatives of the two first coordinates n_1 and n_2 of normals after they have been transformed to camera space. As our goal is to extract continuous curvedness information, we compute such a derivative not by considering an infinitesimal neighborhood, but by differentiating in an extended neighborhood. To this end, we convolve each of the normal coordinate images with Gaussian derivatives:

$$\begin{aligned} \nabla n_1(x, y) &= \begin{bmatrix} n_1(x, y) \otimes g_x(x, y, \sigma) \\ n_1(x, y) \otimes g_y(x, y, \sigma) \end{bmatrix} \\ \nabla n_2(x, y) &= \begin{bmatrix} n_2(x, y) \otimes g_x(x, y, \sigma) \\ n_2(x, y) \otimes g_y(x, y, \sigma) \end{bmatrix} \end{aligned}$$

where $g_x(x, y, \sigma)$ and $g_y(x, y, \sigma)$ correspond to Gaussian derivative kernels in the $\mathbf{x}$ and $\mathbf{y}$ directions respectively, and σ determines the scale at which differentiation is performed (see [tHR03] for further details). In practice, we again make use of the separability of the Gaussian derivative operator ($g_x(x, y, \sigma) = g'(x, \sigma)g(y, \sigma)$ and $g_y(x, y, \sigma) = g(x, \sigma)g'(y, \sigma)$).

The last step needed to obtain L from normal variations is to combine ∇n_1 and ∇n_2 . We take inspiration from previous work that faced the general problem of computing the gradient of a multi-valued image. In the work of Di Zenzo [Zen86], a general solution is presented, that has been used for instance to compute color gradients from $(\nabla R, \nabla G, \nabla B)$ color triplets. We apply it to our normal variations $(\nabla n_1, \nabla n_2)$, and describe the process in the following. The method first computes directional

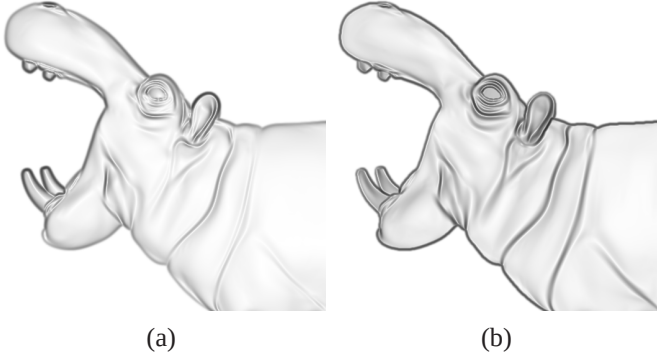


Fig. 2.13: Comparison of curvedness computations: (a) only using the gradient of surface slant for curvedness misses many salient shape features. (b) In contrast, our curvedness attribute takes into account the entire set of shape features.

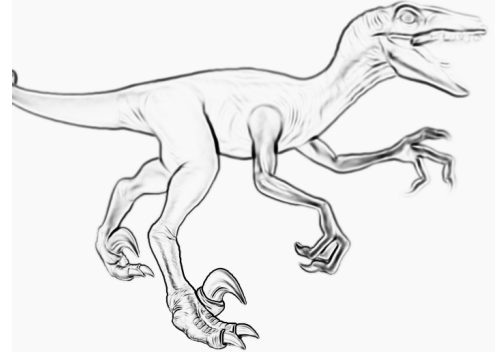


Fig. 2.14: Levels-of-detail effects: Varying scale per pixel provides a way to easily create LODs. In this example, we focus on the raptor's front leg, but any spatial function may actually be used.

gradients in vector space:

$$\nabla n_x = \begin{bmatrix} \mathbf{T}(\nabla n_1)\mathbf{x} \\ \mathbf{T}(\nabla n_2)\mathbf{x} \end{bmatrix} \text{ and } \nabla n_y = \begin{bmatrix} \mathbf{T}(\nabla n_1)\mathbf{y} \\ \mathbf{T}(\nabla n_2)\mathbf{y} \end{bmatrix},$$

where $\mathbf{T}$ denotes the transpose operator. Then, it builds the symmetric tensor field $\mathbf{N}$ defined by

$$\mathbf{N} = \begin{bmatrix} N_{11} & N_{12} \\ N_{21} & N_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{T}(\nabla n_x)\nabla n_x & \mathbf{T}(\nabla n_x)\nabla n_y \\ \mathbf{T}(\nabla n_y)\nabla n_x & \mathbf{T}(\nabla n_y)\nabla n_y \end{bmatrix}.$$

In [Zen86], it is shown that the combined gradient corresponds to the maximum absolute eigenvalue of $\mathbf{N}$. We thus perform a Principal Component Analysis and set L equal to the computed maximum absolute eigenvalue.

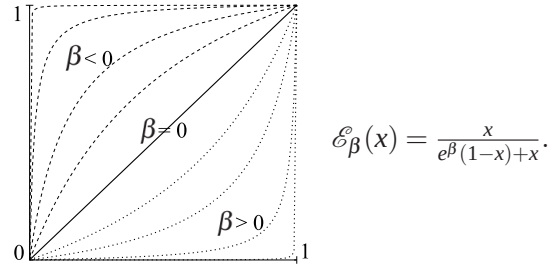
The resulting curvedness attribute identifies the most salient shape features of the depicted object, including the ones missed using a derivative of surface slant (see Figure 2.13 for a comparison).

2.2.4 Combining Measures

In previous sections, we explained how to extract $\mathbf{D}$ and L at a given scale σ . The reason for using the same scale for both measures is to ensure that the convexity information revealed in smoothly varying curvedness areas is similarly smooth. We must also take care when $|\mathbf{K}| = 0$ causing $\mathbf{D}$ to be undefined. This is not an issue as long as $L = 0$, therefore we impose it as a constraint wherever $\mathbf{D}$ is undefined. In practice we use a smooth transition from L to $|\mathbf{K}|$ near $|\mathbf{K}| = 0$ to avoid discontinuities. The resulting vector $\mathbf{S} = L\mathbf{D}$ is our Apparent Relief Descriptor (see Figure 2.12).

In addition, the scale parameter may be used to control the spatial smoothness of our descriptor. We provide two different approaches: it may be set globally for the whole image, and controlled by users; or it may be varied spatially per pixel using an arbitrary function such as depth or a point of focus around the mouse (see Figure 2.14). To implement the spatially-varying approach, one need to compute L at multiple scales and combine them linearly, which is easily done on modern graphics hardware using multiple passes and still runs in real-time. Finally, we also allow users to control L

using the emphasis function $\mathcal{E}_\beta(x) : [0, 1] \rightarrow [0, 1]$. Based on the rational polynomials [Sch94b], the following function exaggerates or reduces local variations:



This function is controlled via a parameter β , which gives comprehensible variations for exaggeration ($\beta > 0$) as well as attenuation ($\beta < 0$).

2.2.5 Application to Different Shading Styles

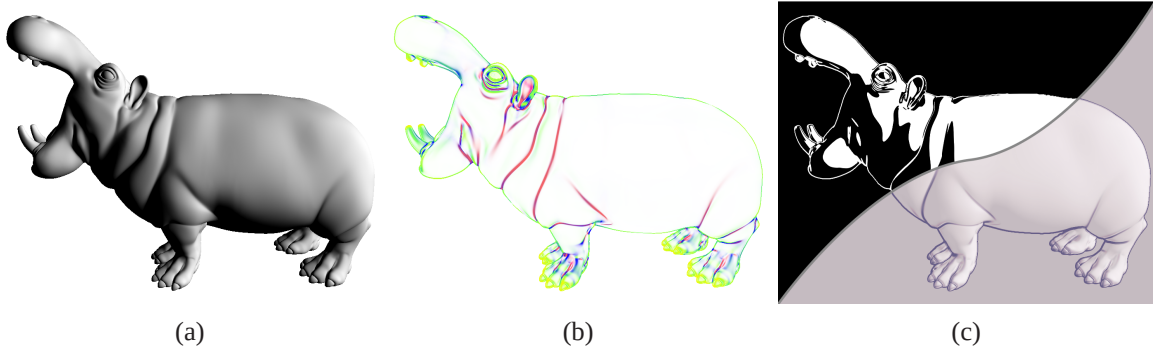


Fig. 2.15: Depicting shape through shading: Starting from a 3D model (a), we extract our view-dependent shape descriptor (b) - (see Figure 2.11 for color code). We then manipulate shading based on *continuous* shape information selected using the descriptor, and create various styles such as minimal shading and cartoon shading (c).

To illustrate the benefits of our Apparent Relief Descriptor, we explain in detail how it is used to depict shape with four different shading styles: cel shading, cartoon shading, minimal shading and exaggerated shading. The overall approach is to create a texture that we call an *Apparent Relief Map* (ARM), which assigns a *relief value* $r \in [0, 1]$ to every possible shape vector. For a given pixel, we then simply use $\mathbf{S}$ as a texture coordinate to lookup a relief value in the ARM. This texture is a straightforward way to choose which shape features are selected as shown in the next Section. An example is given in Figure 2.11-(b): here, everything but planar- and saddle-like surface regions is selected. All the textures in this paper have been done by hand in conventional image processing software.

All our examples run in real-time on modern graphics hardware. Mesh sizes and frame-rates are given in figure captions, and are similar for most styles, except for exaggerated shading that requires more computations.

Cel-shading (Figure 2.16) is the simplest of the four shading styles we present. The idea is to make direct use of relief values from a given ARM as pixel colors. Because this style clearly shows the effect of the chosen ARM, we illustrate it using four different textures. The top-most one simply filters out every surface region that is planar. The ones below provide a finer control over which shape details should be conveyed. The resulting shadings give compelling contour renderings.

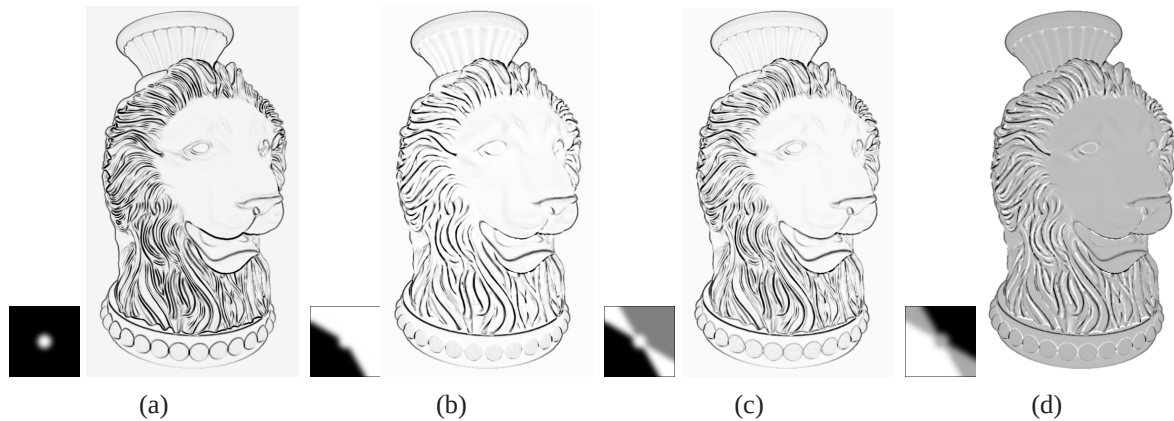


Fig. 2.16: Cel-shading: By simply displaying relief values, we show the versatility of using an ARM as a control interface. In (a), we filter only planar regions; in (b), we keep only convexities. (c) shows an intermediate result between (a) and (b), where concavities are displayed in light gray. As a last example, (d) renders concavities in white, convexities in black, and planar regions in gray (0.4 Mtri / 244 fps).

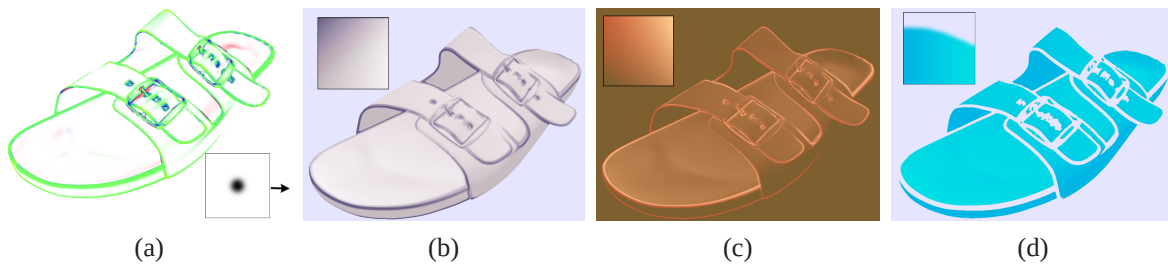


Fig. 2.17: Cartoon-shading: The apparent relief descriptor (a) is used to select convex and concave regions and to lookup into X-Toon textures. In (b), we use a smooth color gradation that gradually fades out in lit regions. In (c), we reverse lights and darks, and show shading variations *inside* shape features. In (d), we use a very subtle color gradient, and trim the alpha channel when the relief value is high, removing shape details and flattening the result (0.3 Mtri / 250 fps).

Conventional **cartoon-shading** [Dec96] may easily be modified to take into account our shape descriptor (see Figure 2.17). The easiest approach is to employ the X-Toon shader of Barla et al. [BTM06]: the classic 1D toon texture is then extended to a 2D toon texture, where the vertical axis corresponds, in our case, to relief values. To illustrate this shading style, we use the ARM of Figure 2.16-(c). As shown in Figure 2.17, X-toon allows users to create more complex shadings, such as smooth color variations located only in dark regions, and removal of shape details done by trimming the alpha channel around strong relief values.

Another place where a shape descriptor is needed is when using drastic shading styles, such as with what we call **minimal-shading**. A good example of this shading style is the appearance of the recent feature-length movie “Renaissance” [Vol06]. Mostly black and white colors are used, so that artists often need to carefully tweak light positions by hand to reveal objects’ shape. This is because shape features are not always visible under conventional lighting, and they need to be reintroduced in some way, an unintuitive and time consuming process. The recent paper of DeCarlo et al. [DR07] proposes to extract highlight lines for reintroducing sharp shape features in dark regions. With our shape descriptor, we rather reintroduce continuous shape information in such a black and white shading style (see Figure 2.18): the idea is to treat relief values obtained from the ARM as contrast values. Minimal shading is thus defined by a simple linear interpolation between a diffuse

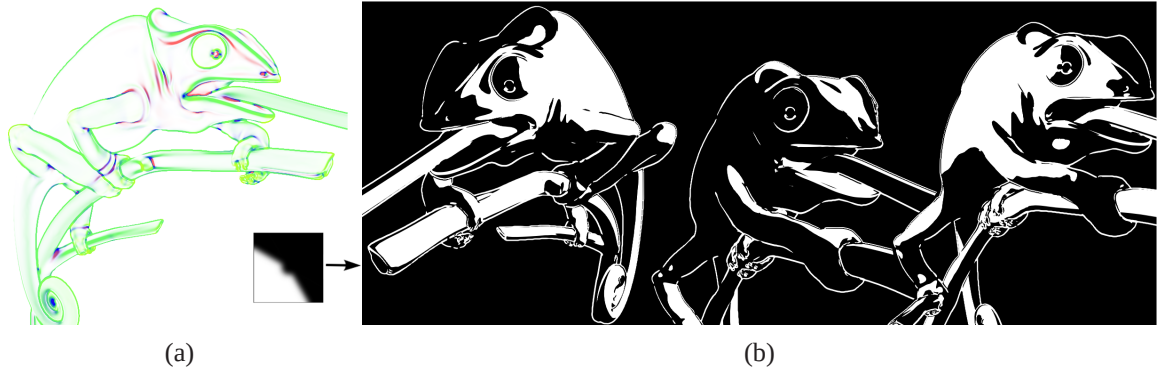


Fig. 2.18: Minimal-shading: The apparent relief descriptor (a) allows to reintroduce the contrast lost using drastically simple shading. Convex regions are used to make shape features appear in black-on-white, or white-on-black: in (b), from left to right, we first rotate around the object to show its dark side, and then move the light around (0.15 Mtri / 250 fps).

and inverted diffuse intensity, using relief as the interpolation parameter: $I = (1 - r)^T \mathbf{n} \mathbf{l} + r^T \mathbf{1} - \mathbf{n} \mathbf{l}$, where $\mathbf{n}$ and $\mathbf{l}$ define the current point's normal and light unit vectors. I is then thresholded at 0.5. To illustrate this style, we use an ARM that filters out everything but convexities.

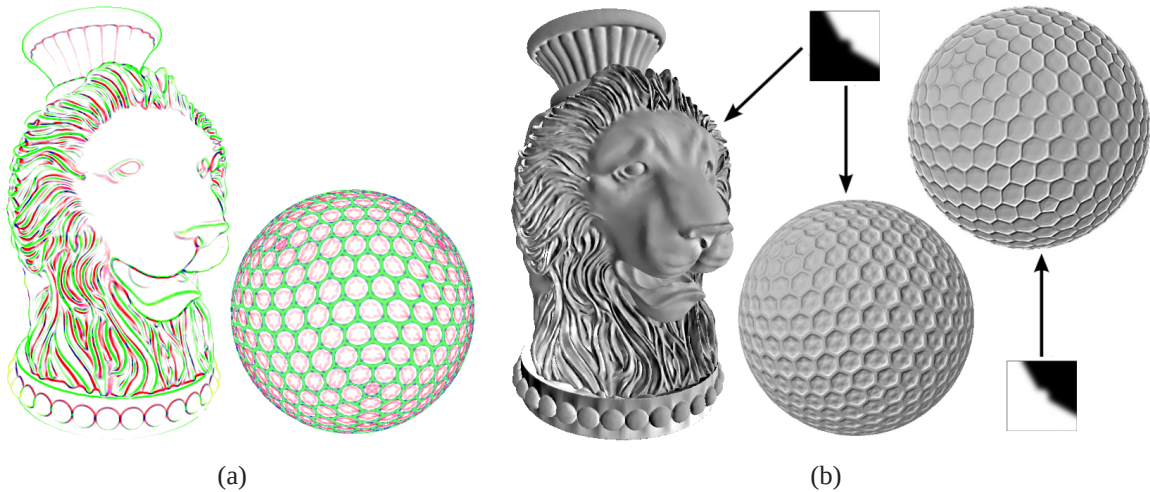


Fig. 2.19: Exaggerated-shading: The apparent relief descriptor (a) may be used to control exaggerated shading. Relief values govern the amount of exaggerated details in specific regions: we select concavities for all the images, and convexities for the right-most image (1 Mtri / 40 fps).

Our last example shows how to control **exaggerated shading** [RBD06] with our shape descriptor (see Figure 2.19). Its main principle is to build a multi-scale analysis of object normals, in the spirit of a Gaussian Pyramid: at each level, normals are averaged to yield a coarser description. Then the authors apply a cosine shading at each level $i = 0..N$ using each time the light direction projected onto the plane defined by the normal in the coarser level. This has the effect of emphasizing local details at each level. Finally, the resulting lighting calculations are linearly combined using per-level weights k_i . We propose to use our apparent relief descriptor to accurately control the location of details that users' wishes to exaggerate. Our approach is to compute the weights according to relief values: $k_i = \mathcal{E}_\beta(r)$, where $\mathcal{E}_\beta$ is the emphasis function, and $\beta \propto N - i/N$. Intuitively, this choice of weights progressively brings in finer and finer details with increasing relief. With our approach we can selectively enhance

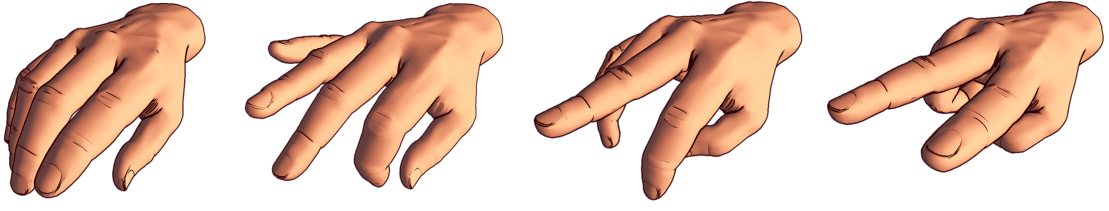


Fig. 2.20: Animated example: Four frames of an animated object, rendered using a cartoon shading style. Note the fine wrinkles that appear on fingers.

convexities or concavities as shown in Figure 2.19.

Many other ARMs and style parameters could be used, and we show additional results in the supplemental videos of [VBGS08a, VBGS08b]. Moreover, the method may be applied to animated objects, provided principal curvatures are pre-computed at each vertex for every frame of the animation. These videos also demonstrate our shading styles on such input as well as Figure 2.20, which shows frames of an animated 3D object rendered in a cartoon style.

2.2.6 Simple User-Controls

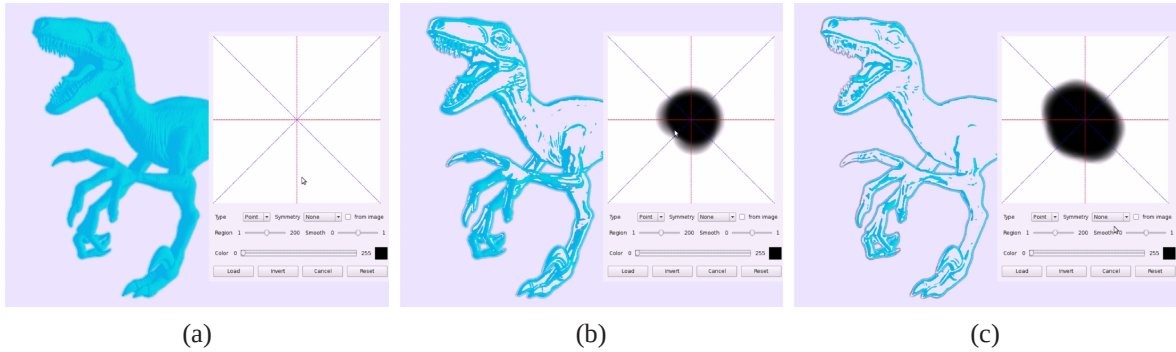


Fig. 2.21: Features selection by ARM painting: After loading an object (the raptor) and selecting a rendering style (here a cartoon-shading), all the features are equally displayed (a). By painting the center of the ARM in black, users remove the quasi-planar regions of the objects (b)-(c) since their weight is set to zero.

As shown in Figure 2.11, the shape vector domain is a comprehensible parameterization of the curvature-related features: caps, cups, saddles, convex, concave and planar regions are organized as section of the 2D domains; the central region corresponds to quasi-planar section of the surface while the boundaries correspond to highly curved ones. Moreover, thanks to our 2D Apparent Relief Map, features are easily weighted in and out using simple interactions.

The first one is a simple painting interface. Users directly paint the weights of the different curvature-based features in the ARM (cf. Figure 2.21). Since our system is fully interactive, they directly visualize the result on the 3D model. For an even more intuitive interaction, it is also possible to directly select the features on the 3D object (cf. Figure 2.22): from the object region, the system extracts the curvature information that corresponds directly to 2D coordinates in the ARM. The weights are thus drawn in the corresponding ARM parts. This control is global: all the similar features are also similarly revealed or hidden.

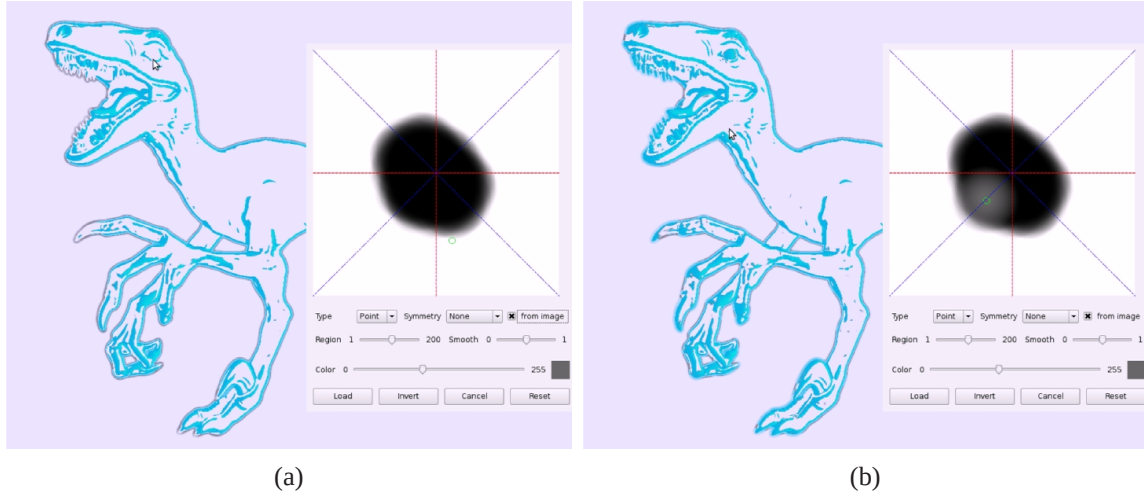


Fig. 2.22: Direct selection of features: To precisely adjust an ARM, users directly select a feature on the 3D object. Using a gray-level for the ARM (a), users select the raptor’s eye, and the corresponding area of the ARM is painting in gray (b). Consequently, the eye and all similar features are revealed since their weight is no more zero.

2.2.7 Discussion and Future Work

Our shape descriptor, called Apparent Relief, makes use of both object-space and image-space attributes to extract continuous information of convexity and curvedness respectively. It provides a flexible approach to the selection shape features, and thus is efficiently used to depict shape through many different shading styles. On top of these benefits, the proposed method provides automatic LOD functionalities that enable a variety of new effects, works in real-time on modern graphics hardware, and is simple to implement. Furthermore, our shape descriptor shows that 2D and 3D analysis can be combined to provide users with comprehensible 2D information coherent with the underlying 3D object.

To illustrate its potential, we have demonstrated its use with four shading styles, including intuitive approaches to control minimal shading and exaggerated shading. Any salient shape feature identified via our descriptor is selected or filtered out via an Apparent Relief Map, and this tool is easy to manipulate by painting or selection interactions. Further shading styles, from realism to non-photorealism, have to be investigated.

There are some limitations to our approach that have to be addressed in future. For object-space analysis, while we can deal with animated scenes, dynamic scenes (e.g., live character animations or natural phenomena) are not tractable because of the costly estimation of curvatures that need to be recomputed at each frame. Secondly, for manifold objects, creating a coherent vector field or principal curvature directions is a complex task related to mesh parameterization [ACSD⁺03]. Note that these limitations are also limitations of most object-based approaches.

For image-space analysis, while controlling the scale parameter enables many interesting LOD behaviors and user-controls, it would be even more interesting to design an automatic scale selection mechanism. For this reason, Scale Space Theory [tHR03] has to be further investigated. Moreover, the current implementation of curvedness extraction is based on an ad-hoc Gaussian filtering, and do not preserve any existing discontinuities that participate to the shape depiction. Improved image filtering techniques have to be investigated to provide a more robust descriptor.

Finally, the current combination of image-based and object-based analyses is a first step to more

versatile one. Currently, the whole process is adapted for a visualization system: it reveals only the features that exist at the current image resolution. This is similar to classical image analysis improved by some 3D information. Some adaptations have to be done for an exploration system, where even some hidden features would be presented to the user. To move forward the exploration context, 3D information and tracking is even more important. As an example, the current user-selection of features is done globally. If a spatial adaptation is possible on the image by a local variation of the descriptor scale based on user-focus or on depth (cf. Section 2.2.3), the relation between this parameter variation and the related part of the 3D object is lost at each camera movement. This parameter adaptation has to be thus attached directly to the 3D object. Moreover, relations between image-space and object-space information issued from the analysis (e.g., curvatures, silhouettes, scale ...) have to be more closely investigated.

2.3 Conclusion

In this chapter, we have introduced new techniques for controlling the appearance of 3D objects: a new BRDF model and its modeling tools, and a new analysis to extract shape characteristics in order to preserve them in expressive shading techniques.

Our BRDF model is defined to allow sketching and painting metaphors for BRDF design. A lobe of our BRDF is represented with a polar curve for its shape and a texture for its color gradient and for precise lobe-shape enhancement. Our representation allows real-time edition of BRDFs, and provides more editing and creative freedom than previous approaches, from realistic to non-photorealistic BRDFs.

Our shape descriptor is based on a combination of object-space and image-space techniques to extract continuous information of convexity and curvedness respectively. It provides a flexible and real-time approach to the selection of shape features, and is efficiently used to re-introduce them in different shading styles. Furthermore, it shows that 2D and 3D analysis can be combined to provide users with comprehensible 2D information coherent with the underlying 3D object.

Their main common limitation is that the presented approaches provide users only with global controls. To improve the user-freedom, we have to develop local edition to precisely adjust all the parameters directly on 3D objects: this would result in spatial variations of these parameters. With apparent relief, we provide a first solution to this problem, but limited to neighborhood defined in image space. For dynamic scenes and moving objects, or for interactive visualization, a real 3D variation of parameters has to be defined.

Our BRDF representation is also mainly limited (cf. Section 2.1.5 for more discussions) by the use of parametric curves as definition of their shape. We have also shown that using image-based masks can reduce such limitation but without totally removing it. We believe that pure texture-based approaches would provide more flexible solutions. To achieve such a goal, the constraints of plausible shading have to be carefully defined in the context of material edition for realistic rendering.

Our shape descriptor currently does not take into account some of the shape discontinuities: issued from visibility events from viewpoint (silhouettes, depth discontinuities ...) or sharp 3D features, our image space filtering have to be bounded by those curves for an improved accuracy. We have introduced recently a first solution [VPB⁺09], but the integration of discontinuities as constraints has to be improved. An accurate and robust combination of 3D and 2D analysis will participate to a general improvement of view-dependent shape depiction (cf. Section 2.2.7 for other possible extensions). This hybridization is even more important than for sketching, since perception generally spans in image space and shading largely participate to our perception of 3D shapes.

Both techniques introduce new controls of 3D object appearance due to direct shading, with their advantages and limitations discussed in the previous Sections. For more realistic rendering, either more complex shading (e.g., with real BRDFs, soft shadows ...) or global illumination have to be investigated for an improved user experience. For such a goal, a high-level understanding of the global behavior of light propagation is needed while still providing more intuitive local editions. Similarly to the combination of 2D interactions for 3D scenes, better knowledge on how these complex global phenomena can be locally abstracted is required to develop simpler tools. In the next chapter, we propose a first step for one possible solution to this problem.

In the previous chapters, we have introduced new tools for local definition of a 3D scene: free-form modeling of 3D objects in Chapter 1 and their local definition of appearance in Chapter 2. When it comes to a complete scene, the interactions between its different components have to be also defined: this requires more high-level and global user-control. For lighting, this means considering all the multiple inter-reflections and shadowing effects that occur.

As said in the introduction, as soon as the computational power has reached a sufficient level, a lot of research has been done in trying to simulate the reality. Over all the different aspects to reach this goal, from static geometry to animated scenes, illumination has raised a lot of interest. This is naturally due to the fact that for most of us, the vision is our main way to perceive the environment. First focusing on physical accuracy to produce convincing images, most of the researches are now based on the research of powerful approximations to reach visual plausibility for interactive rendering.

Intuitive controls for global illumination have only recently emerged as research domain [BAERD08, OKP⁺08]. In this chapter, we introduce a new representation of indirect lighting based on vector quantities developed bearing in mind future and possibly more user-friendly editions. This work is for us a perfect illustration of the fact that developments based on a user-friendliness point of view can result in more generic and versatile solutions. The representation and its applications presented in this chapter has been validated by the following publications: [PRG⁺08, PRL⁺08]

3.1 Previous Work

Researches in global illumination have resulted in a very large number of publications: a full summary of previous work is out of the scope of this document. Readers can refer to the book of Dutré et al. [DBB06] for a detailed presentation of this domain. In this section, we present a short overview of the developed techniques, with a particular focus on the possible user-controls.

3.1.1 Definitions

The global illumination problem has been formalized by the well-known *Rendering Equation* [Kaj86]. For each wavelength ($\lambda \in [380, 780]$ nm for visible light), this equation simply expresses a steady state

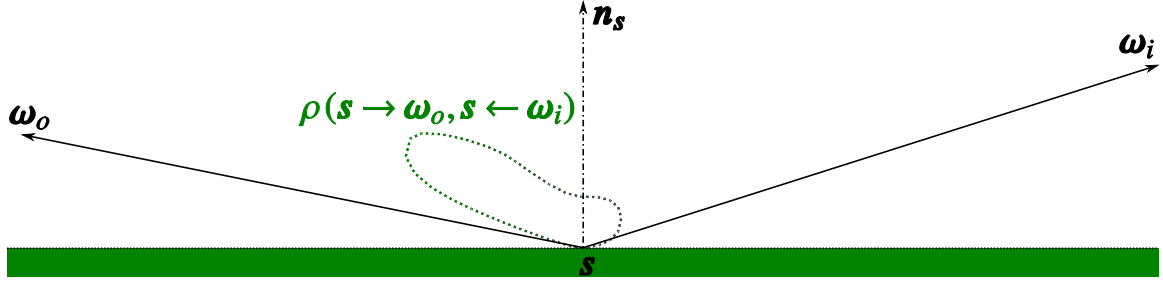


Fig. 3.1: Geometric configuration of a reflection.

of light propagation in a non-participating environment¹: the entire emitted energy from a point $\mathbf{s}$ of portion of surface and around a direction $\mathbf{w}_o$ (this quantity is called *radiance*: Watt per square meter and per steradian that is, $W.m^{-2}.sr^{-1}$) is the sum of the self-emitted energy (for light-sources) and of the reflected incoming energy:

$$L(\lambda, \mathbf{s} \rightarrow \mathbf{w}_o) = L_e(\lambda, \mathbf{s} \rightarrow \mathbf{w}_o) + \int_{\Omega} \rho(\lambda, \mathbf{s} \rightarrow \mathbf{w}_o, \mathbf{s} \leftarrow \mathbf{w}_i) {}^T \mathbf{w}_i \mathbf{n}_s L(\lambda, \mathbf{s} \leftarrow \mathbf{w}_i) d\mathbf{w}_i.$$

In this equation, and in all the followings, we use the same notation (cf. Figure 3.1 and [DBB06]): ρ stands for the Bidirectional Reflectance Function (BRDF, as detailed in Section 2.1) or more generally, the Bidirectional Scattering Function (BSDF [Hec91]), $\mathbf{w}_o$ (resp. $\mathbf{w}_i$) stands for the out-going or view direction (resp. incident direction), Ω denotes the whole space of possible directions, $L(\lambda, \mathbf{s} \rightarrow \mathbf{w})$ stands for the emitted energy from $\mathbf{s}$ in the direction $\mathbf{w}$ and $L(\lambda, \mathbf{s} \leftarrow \mathbf{w})$ for the received energy at $\mathbf{s}$ from the direction $\mathbf{w}$. Finally, ${}^T \mathbf{w}_i \mathbf{n}_s$ denotes the dot product between the normal $\mathbf{n}_s$ at point $\mathbf{s}$ and the direction $\mathbf{w}_i$.

In Computer Graphics, peoples generally do not solve the rendering equation for the full spectrum of visible light for efficiency purpose. Instead, this equation is evaluated for colored values of radiance, using XYZ of RGB color spaces [CIE]. For each color component, we approximate the light propagation by using the same previous equation

$$L(\mathbf{s} \rightarrow \mathbf{w}_o) = L_e(\mathbf{s} \rightarrow \mathbf{w}_o) + \int_{\Omega} \rho(\mathbf{s} \rightarrow \mathbf{w}_o, \mathbf{s} \leftarrow \mathbf{w}_i) {}^T \mathbf{w}_i \mathbf{n}_s L(\mathbf{s} \leftarrow \mathbf{w}_i) d\mathbf{w}_i. \quad (3.1)$$

This approximation does not introduce any error only in the case of white incident lighting [Bor91] (i.e., the incident intensity is similar for each wavelength). Discussions about this approximation and some proposed user-controlled corrections are presented in Appendix A.

In lambertian (or diffuse) environments where the BRDF is independent on the view and incident directions (i.e., $\rho(\mathbf{s} \rightarrow \mathbf{w}_o, \mathbf{s} \leftarrow \mathbf{w}_i) = \rho_d$), the rendering equation is simplified using a new radiometric value: the *radiosity*. Introduced in Computer Graphics by Goral et al. [GTGB84], radiosity represents the whole energy emitted in all directions in a visible hemisphere from a portion of surface: Watt per square meter - $W.m^{-2}$. Using albedo $\tau = \pi \rho_d$ instead of BRDF, the radiosity equation is

$$B(\mathbf{s}) = \int_{\Omega} L(\mathbf{s} \leftarrow \mathbf{w}_i) h(\mathbf{n}_s, \mathbf{w}_i) d\mathbf{w}_i = B_e(\mathbf{s}) + \tau I(\mathbf{s}),$$

where $I(\mathbf{s})$ is the *irradiance* that is, the average energy received per square meter at surface point $\mathbf{s}$ and $h(\mathbf{n}_s, \mathbf{w}_i)$ the hemisphere function as defined in Appendix B and Section B.1.1. This radiometric

¹no scattering events are taken into account during the light propagation in the air

quantity is expressed as

$$I(\mathbf{s}) = \int_{\Omega} \mathbf{T} \omega_i \mathbf{n}_s L(\mathbf{s} \leftarrow \omega_i) h(\mathbf{n}_s, \omega_i) d\omega_i = \int_{\Omega} \mathbf{T} \omega_i \mathbf{n}_s B \circ \mathbf{p}(\mathbf{s}, \omega_i) h(\mathbf{n}_s, \omega_i) d\omega_i, \quad (3.2)$$

where $\mathbf{p}(\mathbf{s}, \omega_i)$ is a function that returns the first visible point from $\mathbf{s}$ in direction ω_i .

All these equations are recursive and they express all possible light paths in a 3D scene. Their evaluations result in the simulation of complex lighting phenomena but they are generally time-consuming. By using a Fourier analysis [DHS⁺05] or first order approximation [RMB07], theoretical studies have identified some of the most complex lighting phenomena. Unfortunately, the results of these analyses are not fully comprehensible by standard users since they require a solid mathematical background. In order to describe the different visible lighting effects, the regular expression for light paths, introduced by Heckbert [Hec90] and improved by Suykens [Suy02], is more intuitive. In these expressions, L denotes a position on a light source, D a diffuse reflection, S and G a specular and glossy reflections, and E a position on the final receptor (e.g., the eye). All light paths are thus characterized by $L(D|G|S)^*E$ where “*” stands for zero or multiple occurrences and “|” for the “or” operator. Similar idea has been used for compositing different lighting simulations in [SSH⁺98]. The different components of global illumination are thus divided in, as examples:

- *Direct*: reflection of light directly issued from light sources - $L(D|G|S)E$
- *Indirect*: lighting due to reflected light sources - $L(D|G|S)^+E$
- *Diffuse environment*: the solution for such an environment is view-independent and thus does not requires a view-point - LD^*
- *Diffuse solution*: such a solution contains multiple non-diffuse reflections but is still view-independent - $L(D|G|S)^*D^*$
- *View-dependent reflections*: this paths contains all the glossy and specular paths visible from a viewpoint - $(L|D)(G|S)^+E$

3.1.2 Algorithms

Computer Graphics relies on fast algorithms to generate visually correct images. Over more than 20 years, a large number of techniques have been introduced to solve these different equations.

Radiosity Techniques

With its diffuse assumption, radiosity is direction-independent quantity. The radiosity equation can thus be solved using confidently classical finite element methods based on a simple discretization of a 3D scene. From its introduction in Computer Graphics by Goral et al. [GTGB84], such techniques have contributing to the raise of global illumination in image synthesis. A complete overview of the different techniques prior to year 1994 can be found in the books of Sillion and Puech [SP94] and Ashdown [Ash94].

From the original and classical finite element methods, the research in efficient computation has resulted in a large number of new approaches. To reduce the original cost, some solutions adjust the accuracy according to its contribution to the final image [SAS92], but such approaches are view-dependent. For a more generic approach, hierarchical and thus wavelet [GSCH93, SGCH94, HCA00, CAH00] representations have been introduced: from original $\mathcal{O}(N^2)$ cost (where N is the number of discretization elements), they lead to a $\mathcal{O}(K^2 + N)$ cost while preserving the confidence due to finite elements (K denotes the initial number of object in the scene before discretization). To further reduce the complexity, the initial number of object has to be also reduced. This is achieved by using

clustering [SAG94, SDS95] that make use of a volumetric approximation of a collection of 3D objects, or that make use of mesh simplification [WHG99, DB00]. Such approaches increase the efficiency in computing plausible solutions on complex 3D scenes while reducing the control on the resulting physical error [HDS99]. Similar problems exist for the extension of radiosity techniques to more generic surfaces such as points [DYN04], implicit surfaces [MAP00a], or techniques based on more generic basis functions [MAP00b].

Despite their success and their importance in the history of Computer Graphics, all these techniques are limited to diffuse environments. Three kinds of improvements have been proposed in order to extend the computation to more general contexts. The first ones combine finite-element for the diffuse part of a global illumination solution with ray-tracing and derived techniques (cf. the following sections dedicated to stochastic integration schemes): the non-diffuse reflections are either added in a second step after convergence of the finite element solution [WCG87] or directly merged into the iterative solver [CRMT91, GDW00]. While such approaches provide more or less easy combination for an elegant solution for general environments, they also increase the number of parameters due to the use of different techniques. The resulting parameters are generally not completely independent and thus not fully comprehensible. The second is to replace the simple direction-independent radiosity value by a directional distribution of outgoing energy (i.e., radiance) [SAWG91, SDS95, SSG⁺00]. The third solution relies on an implicit directional representation of incident energy [SH94, SS98, DBG99]. These techniques compute the energy transfer between each discretization patch and rely of a gathering step to compute the final reflected energy toward the image. Unfortunately, the last two kinds of approaches increase drastically the amount of required storage or processing time.

Radiosity techniques inherit their advantages from finite element methods. Firstly, they take benefit of the good reputation of such techniques due to a long history of developments and uses in physical simulations: the final accuracy is theoretically easy to control. Secondly, they provide a view-independent solution on the whole 3D scene, even in the case of stochastic integration of the radiosity equation [Sbe97]: this is well-suited for physical simulations and their visualization. Nevertheless, in the case of hierarchical solutions based on clustering, the resulting errors are hardly bounded [GH96, SSS97]. Researches have still to be done to be fully usable.

Unfortunately, all these techniques inherit also their disadvantages from finite element methods. Firstly, they require a whole discretization of the 3D scenes, leading to a storage requirement dependent on the scene complexity. With the increasing richness of virtual world, the memory requirement can simply be prohibitive, even for memory efficient solution [SSSS98, GD99]. Secondly, the main control provided to users is on the resulting accuracy. In Computer Graphics, qualitative errors combined with simple user-controls are often preferred to quantitative ones: this leads to faster solutions and more comprehensible parameters.

Stochastic Integration

The most successful techniques developed during the past 20 years are based on stochastic integration. Indeed, the convergence of such schemes is independent of the complexity of the integration domain: in global illumination, this domain is the set of all the possible light paths in 3D scene. Furthermore, their minimal requirement are simple and versatile computational kernels that contain only 3D intersections with rays, 3D data structures for enhanced performance in estimating 3D intersections, and evaluation of light reflection at the intersections. An overview of efficient implementations is provided in the books of Glassner [Gla89b] and Sherley [She03]. As an example of their versatility, such computation is even available for recent point-based representation of 3D surface [SW00, AA03].

There are two main families of algorithms. The first ones are based on ray-tracing. Their principle

is quite simple since they are all based on sampling the integration domain by randomly selecting a set of representative light paths, and on the evaluation of light propagation along these samples. A complete overview is presented in the book of Dutré et al. [DBB06]. Such an approach has been used for solving the rendering equation as introduced by Kajiya [Kaj86]: a ray issued from the view-point is propagated in the scene. At each intersection, the direction is reflected and the direct lighting is evaluated. This simple algorithm is called *path-tracing*. Since the rays are issued from the viewpoint, lighting phenomena that highly dependent on glossy or specular reflection of light sources are hardly detected ($L(G|S)^*$ paths). To overcome this limitation, one possibility is to emit rays from both light sources and the viewpoint: such an approach is called *bidirectional ray-tracing* [LW93, VG94, Laf96]. To achieve a faster convergence, local adaptation of paths can be done: such techniques are called *Metropolis light transport* [VG97, Ve97].

Despite their versatility, all these techniques are prone to stochastic noise. To main control to reduce this artefact is to increase the number of samples. This control is simple, but in general, the number of sample has to increase drastically to reduce the noise since these methods converge in $\mathcal{O}(\sqrt{N})$ where N is the number of samples. Thanks to Metropolis light transport, users can tune the ray-mutation strategy to emphasize some effects and increase the convergence to a noise-less solution. This control requires a good understanding of stochastic integration.

As previously said, in Computer Graphics, qualitative errors are often preferred to quantitative ones. For stochastic integration, we generally prefer bias to noise: the converged solution can slightly get away to the rendering equation while preserving the general behavior. On example of such bias is a post-filtering step for noise removal [McC99].

The control on bias is one of the reasons of the success for the second family of stochastic methods: *particle tracing* [Wan01] combined with *density estimation* [Wal98]. Their principle is similar to ray-tracing: path samples are issued from light sources and propagated into the environment. The main difference is that first all the intersection with 3D environment of the path samples are stored (such an intersection is called a particle or a photon) and second, the lighting intensity is estimated based on the density of particles. Heckbert [Hec90] use textures for encoding the illumination: the energy received in each texel is the sum of the energy of the particles that have intersected this texel. Other techniques [WHSG97, Mys97] estimate this illumination directly on each patch of a 3D scene discretization. Unfortunately, mesh-based density estimation suffers from the same limitations than radiosity techniques: they requires a discretization of the 3D scene. The most successful techniques introduced by Wann Jensen [Wan01] is called *photon mapping*. It relies on both a kernel-based density estimation that allows to locally adapt the size of the kernel to the particle density, and also on a final reconstruction based on ray-tracing to compute the reflections toward the image and to add all view-dependent reflections. Thanks to this final step, the bias on the density estimation is masked by a highly accurate final reflection. This approach computes very efficiently low-frequency illumination like diffuse inter-reflections thanks to density estimation, and high-frequency specular reflections thanks to ray-tracing. Unfortunately, middle-range frequencies such as in glossy reflections are hardly estimated by density estimation and still rely on stochastic integration. Furthermore, the choice of the non-independent parameters for the density estimation is not user-friendly, and is difficult to estimate automatically [Chr99].

With some well-tuned algorithms [DBMS02], interactive stochastic integration is possible. But in most cases, they are still computational expensive. Optimizations rely in general on caching some information in order to factorize similar paths [WRC88, CB04]. Indeed, paths issued from light sources and low-frequency illuminations are efficiently estimated and easily cached using spatial interpolation of scalar values such as irradiance, or of directional quantity such as radiance [KBPŽ06]. But, new parameters are again introduced, leading to difficult and empirical user-selection to obtain a good

trade-off between quality and rendering time.

Hardware-Based

One of the main applicative contexts of global illumination is interactive visualization of 3D scenes. In standard interactive applications, common approaches precompute lighting and store it as diffuse lightmaps [Hec90]. However, highly detailed geometries require highly detailed maps to capture all lighting variations. For almost-planar surfaces or slowly varying geometric details, one can use directional lightmaps [HP02] that store, for each texel, a directional light source. This reduces the required resolution but, generally these approaches are only efficient for direct lighting.

Thanks to recent and large increases of computational power of graphics hardware, it is now possible to develop interactive techniques that incorporate global illumination effects [DBB06]. The introduction of *anti-radiance* [DSDD07] removes visibility estimation and is more suitable for hardware rendering. However, it requires a minimal number of passes proportional to the scene depth complexity. Radiosity techniques [CHL04, DSDD07] and mesh-based global illumination [NPG03, NPG05] can be implemented on modern GPUs, by storing the indirect lighting into environment maps [MMP02]. Stochastic solutions have also been implemented on GPUs, such as those based on *photon-mapping* [PDC⁺03] or radiance caching [GKBP05]. Real time is only achieved with *Instant Radiosity* [Kel97, SIP07] combined with incremental techniques [LSK⁺07]. All these techniques rely on fast rendering of complex direct lighting using modern GPUs. But in order to achieve interactivity, some limitations are generally introduced: the number of indirect light bounces is limited (in general to only two), as well as the number of light sources.

3.1.3 Precomputation

Due to the still highly complex computation, global illumination relies more and more on precomputation to speed-up the process and sometimes, to reach interactive frame-rates: precomputation is often the chosen solution for hardware-based visualization of 3D scenes with global illumination.

Caching

For ray-tracing, caching has been introduced by Ward et al. [WRC88] for storing diffuse indirect lighting. The 3D scene is sampled using a criterion dependent on normal variation and on visibility in order to adapt the density according to the illumination variations. The reconstructed illumination is weighted sum of all the cached samples. To reduce this complexity due to the use of reconstruction basis with a global support, cached intensity is estimated using a local neighborhood of cached position, leading to a non-continuous reconstruction even with the introduction of irradiance gradients [WH92]. Caching is also the core of efficient implementation of photon mapping [Wan01]. A well-tailored sampling [KBPŽ06] combined with directional representations of incoming radiance [KGPB05] improves the smoothness but does not lead to a continuous estimation: partition of unity on these non-uniform samples would have to be used. One of the most interesting solutions proposed by Arikan et al. [AFO05] introduces a two-level caching: since radiance due to distant objects exhibits low spatial and angular variation, it is directly estimated with cached data; the radiance due to local geometry is estimated with an indirect access to the cached data by a one-bounce ray-tracing. The main advantage of caching is the resulting speed-up even if it introduces new parameters. Another indirect advantage is that it can be modified in order to fit users' idea of the indirect illumination [TL04].

Precomputed Radiance Transfer

To render interactively complex lighting effects, a whole research domain has emerged recently: *Precomputed Radiance transfer* or PRT. The main aim is to precompute the response of 3D objects to infinitely distant light sources (i.e., directional light sources). Ramamoorthi and Hanrahan [RH01] first introduce this approach for diffuse and direct lighting due to distant environment approximated by an environment map: the lighting response is computed for each vertex of a 3D mesh. PRT has thus been extended to more general lighting effects with multiple inter-reflections [SKS02], to changing viewpoints for non-diffuse objects [LSSS04, WTL04, TS06] and even to normal mapping [Slo06]. The main advantage of such techniques is that users can modify interactively the distance lighting on static objects. Nevertheless, users have no control on the radiance transfer and PRTs are still limited to distant lighting. Near-field lighting for interactive design of globally lighted scenes has been only recently studied [PWL⁺07]. Unfortunately, in most cases, the storage requirement is quite large due to its dependency of the directional accuracy required for highly non-diffuse surfaces, and due to its dependency on geometric complexity. The introduction of clustering with Principal Component Analysis [SHHS03], spectral mesh basis [WZH07] and gradients [AKDS04] reduces this limitation, but does not discard it.

Excepted [PWL⁺07], all these techniques are limited to a single 3D object and to distance lighting. Kristensen et al. [KAMW05] precompute the lighting response for different position of point-light sources: it allows users to design the lighting by selecting the different position of light sources with real-time feed-back. Another solution is to precompute the direct-to-indirect transfer (e.g., Hašan et al. [HPB06]), and rely on a fast evaluation of direct lighting for efficient update (like hardware-based direct lighting). To reach a sufficient visual quality, the sampling strategy needs to depend on the geometric complexity. Thanks to a hierarchical expression, Lehtinen et al. [LZT⁺08] extend this approach to more complex objects and vector representation. Unfortunately, this vector representation is linked to the normal direction of the sample point leading to an interpolation scheme that still relies on a dense sampling for highly detailed objects. Furthermore, due to view-dependent strategy, there is no guarantee that every potentially visible point can have in their neighborhood enough samples to recompute the illumination [RGKS08]. All these solutions are also limited since users can only manipulate light sources.

3.1.4 Energy Representations

The representations of the different quantities involved in global illumination participate to the possibility of user-edition. In this section, we present an overview.

Scalar

The simplest representation generally associated with the diffuse approximation is a scalar value for each patch of a discretization or each sampled position of a 3D scene [WRC88, MAP00a, MAP00b]: radiosity and irradiance. Such representation is also simple to manipulate [OKP⁺08] since it can potentially be directly painted on 3D surfaces.

Directional Distribution

Unfortunately, scalar data are too restricted for general purpose use in global illumination. Furthermore, they are tied to the underlying geometry and thus require a large discretization for highly

detailed objects. For more general purpose, most of the solutions are based on a directional representation of incident radiance, irradiance or implicit visibility [DSDD07]. A very good overview is presented by Lessig and Fiume [LF08].

Issue from researches on polynomial approximations [Arv95] of a directional signal, Spherical Harmonics (e.g., [SAWG91, SDS95, SKS02, KAMW05, Slo06, KL06]) and derivatives such as Hemispherical Harmonics [GKPB04] are widely used thanks to their orthonormality. However, their basis functions have a global support and thus, the resulting representation in frequency domain cannot localize directional lighting phenomena: their user-edition may be less comprehensible than a direct work on the original signal (like editing an image in frequency domain rather directly). Furthermore, they are oriented toward one main axis, emphasizing phenomena along this direction and reducing again the localization of un-aligned ones.

On the other side, wavelets are localized in both space and frequency and therefore are efficient for the representation of radiance [CSSD94, SSG⁺00] and of all-frequency signals [NRH03, NRH04, LSSS04, WTL04]. Unfortunately, on a sphere, they require tailored subdivision schemes (e.g., [SS95, LF08]) and are also often limited to piecewise constant functions: a smooth reconstruction of high-frequency signals is time and memory consuming [SSG⁺00].

Recently, *Spherical Radial Basis Functions* have been introduced [GKMD06, TS06]. SRBFs are localized in both space and frequency but, in contrast to the two previous representations, computing the basis representation of a signal is expensive. Nevertheless, they represent a good trade-off for user-edition of a directional signal.

The work most related to ours is the *Irradiance Volume* [GSHG92] and its extension used by different game engines (e.g., [MFE07]). The original Irradiance Volume is a bi-level regular grid that stores irradiance values at each vertex position. This representation has two main advantages:

1. regular grids can be more easily implemented on GPUs;
2. the volumetric aspect allows to decorrelate the grid resolution from the geometric complexity.

However, the spatial interpolation scheme, to ensure a smooth reconstruction of the irradiance, is more complex than a classical trilinear interpolation scheme used with a regular grid.

Vector Quantities

On one side, scalar values are too tied to the geometric complexity of a scene. On the other side, directional distributions are more complex and resource demanding and thus are more difficult to manipulate. We thus take a look on vector representations.

Introduced by Arvo [Arv94], an *Irradiance Vector* is a vector quantity which norm represents the incident energy over a whole hemisphere and which direction represents the main incident lighting direction. It can thus be associated to a local directional light source, quantity that can be easily manipulated [TL04]. Moreover, it is robust to geometric variations as shown in the works of Lecot et al. [LLAP05], Willmott et al. [WHG99] and Gobbetti et al. [GSA03], and as pointed out by Lehtinen et al. [LZK⁺07, LZT⁺08]. In general, vector representations are more independent on the geometry, as shown by the development of *Light Vectors* [ZSP98, SP01].

3.1.5 Independence on 3D Complexity

On important point to discuss is the independence on the 3D scene complexity. Such a decorrelation increases the independence of parameters between the 3D scenes description and the illumination representation.

With Photometric Independence

Classical caching structures [WRC88, Chr99, CB04]) store irradiance as it allows to change the diffuse albedo of materials without having to recompute the cached values. To overcome the limitation to diffuse reflectance or at best, low-frequency BRDFs imposed by irradiance caching, new schemes based on incident radiance caching have been introduced. Encoding incident radiance by spherical harmonics [KBPŽ06, AFO05] or wavelets [SSG⁺00] is more accurate than constant basis [GSHG92], but with both representations, the number of coefficients quickly explodes for high-frequency BRDFs, thus still limiting the method to moderately glossy BRDFs. A trade-off between the three following points has to be considered.

With Geometric Independence

While the irradiance is a geometric dependent quantity, the radiance is not. Thus the radiance caching theoretically offers geometric robustness. Unfortunately, the usual strategy [WRC88] used to place the precomputed samples depends on the underlying geometry. Therefore, radiance caching cannot be applied in the case of highly detailed surfaces, because the number of samples quickly becomes huge. While light vectors [ZSP98] are also geometrically robust, they are not photometric independent.

With Continuous Reconstruction

Another issue with caching involves the interpolation scheme used during the rendering pass. Irradiance and radiance caching schemes need to store their samples in an efficient structure (kd-tree or octree) in order to quickly retrieve them when interpolation is needed. However, due to the combined facts that these samples are not placed on specific positions and only an interpolation on local neighborhood is performed, these schemes cannot ensure a continuous reconstruction of the stored radiometric quantity. On the contrary, with volumetric representations, such as irradiance volumes [GSHG92], the continuous interpolation is easier to perform. Unfortunately as the irradiance volumes cache incident radiance, numerical integration is required at the rendering step.

3.1.6 User Edition and Conclusion

Editing global illuminated scenes is a complex task due to the multiple inter-reflections. We believe that user-friendliness of global illumination relies more on the possibility of a direct control on the light in order to match it with users' desires than on a careful choice on parameters that control only the accuracy of the simulation.

Some previous solutions offer the possibility to edit 3D scene by moving the 3D objects (in purely diffuse scenes [Sha97, DS97] and with some non-diffuse paths [GD01, DBMS02]): view-dependent reflections would rely on interactive ray-tracing [Shi06]. As described in previous sections, some others rely on precomputation to allow interactive edition of light sources [KAMW05, HPB06]. None of them offer an intuitive tool to freely design the illumination. More recently, Pellacini et al. [PBMF07] have introduced an optimization process to define the parameters of light sources corresponding to a painting solution by users: such solution is limited to direct lighting and can hardly be applied for full global illumination.

Ben-Artzi et al. [BAERD08] introduce a new polynomial representation of BRDFs that enables interactive BRDF editing with global illumination. They use a high-quality representation for the BRDFs at the first bounce from the eye and lower-frequency (often diffuse) versions for further

bounces. The solution is limited to editing the global illumination effects due to BRDFs in on single image. Nevertheless, they show that simple polygonal basis can be sufficient for user-edition purpose.

Obert et al. [OKP⁺08] have shown that, in cinematic lighting design, physical accuracy is less important than careful control of scene appearance. They thus propose a new representation for artistic control of indirect illumination. They encode users' adjustments to indirect lighting as scale and offset coefficients of the transfer operator. These adjustments are done for multiple key-frames and interpolated in-between. Such an approach is limited to linear transformation of lighting operator. We believe that a new representation of indirect lighting can help in introducing more freedom in lighting design.

It is important to notice that these last approaches do not simulate the whole propagation for each modification. They in fact rely on simplified or abstracted representations of the phenomena. This is a requirement to reach an interactive editing.

3.2 Irradiance Vector Grid

After having briefly reviewed previous work for global illumination, we want to add one more remark about lighting design. The most related real-world techniques for lighting edition are the ones use by photographers to emphasize the shape of real objects: they carefully select and position a set of light sources. In Computer Graphics, peoples are used to manipulate point and directional light sources: Tabellion and Lamorlette [TL04] have shown that indirect lighting can be editing similarly than in real-world by carefully select and position a set of local light sources. Simple to implement on graphics hardware, such light sources allow interactive visualization of glossy reflection at interactive frame-rate [WAL⁺97].

Moreover, regarding the three issues discuss in Section 3.1.5, we propose an alternative volumetric data structure based on irradiance vectors that offers improved geometric robustness and similar photometric robustness as other comparable methods. To provide a smooth reconstruction of indirect illumination, we use a continuous interpolation scheme which does not depend on surface geometry. Furthermore, our representation is easy to adapt on GPU and has low memory consumption.

3.2.1 Directional Irradiance Vector

For a given wavelength, the *irradiance vector* $\mathbf{I}(\mathbf{n}_s)$, as introduced by Arvo [Arv94], is defined for a point $\mathbf{s}$ with normal $\mathbf{n}_s$ as

$$\mathbb{R}^3 \times \Omega \rightarrow \mathbb{R}^3 : \mathbf{I}(\mathbf{n}_s) = \int_{\Omega} L(\mathbf{s} \leftarrow \boldsymbol{\omega}_i) \boldsymbol{\omega}_i h(\mathbf{n}_s, \boldsymbol{\omega}_i) d\boldsymbol{\omega}_i, \quad (3.3)$$

using the same notation of Equation 3.2. The irradiance vector stores radiometric and geometric information; it is directly related to the diffusely reflected radiance:

$$L_r(\mathbf{s}) = \frac{\tau}{\pi} \mathbf{I}(\mathbf{n}_s) \mathbf{n}_s \quad (3.4)$$

where τ is the diffuse albedo at point $\mathbf{s}$ and T denotes the transpose operator. The main benefit of irradiance vectors compared to irradiance is that for a local variation of a surface normal, the reflected radiance can be adjusted, making this representation more **geometrically robust**. To evaluate Equation 3.4, one may use any global illumination algorithm such as particle-tracing or path-tracing. Intuitively, an irradiance vector represents the intensity of the incident lighting and the mean direction where it comes from.

Bear in mind that we want to compute the reflected radiance $L_r(\mathbf{s})$, where the normal at $\mathbf{s}$ may be along any direction. Therefore, we need an efficient representation to store the incident illumination for any direction. We thus subdivide the direction space with six overlapping hemispheres, where each hemisphere is oriented toward a main direction $\boldsymbol{\delta} = \pm\mathbf{x}|\pm\mathbf{y}|\pm\mathbf{z}$. We precompute an irradiance vector $\mathbf{I}_{\boldsymbol{\delta}}$ for each of the six hemispheres to represent the incident illumination. With an appropriate interpolation scheme, we combine the different values of $\mathbf{I}_{\boldsymbol{\delta}}$ to evaluate $\mathbf{I}(\mathbf{n}_{\mathbf{s}})$ and thus $L_r(\mathbf{s})$ for any normal $\mathbf{n}_{\mathbf{s}} = (n_x, n_y, n_z)$. The interpolation is only done for three out of the six possible directions of $\boldsymbol{\delta}$. The choice between $\pm\mathbf{x}$ (resp. $\pm\mathbf{y}$ and $\pm\mathbf{z}$) is done according to the sign of n_x (resp. n_y and n_z). The *directional irradiance vector* $\mathbf{I}(\mathbf{n}_{\mathbf{s}})$ is obtained by remapping the three spatially interpolated irradiance vectors according to the normal direction $\mathbf{n}$ at point $\mathbf{s}$:

$$\mathbf{I}(\mathbf{n}_{\mathbf{s}}) = \mathbf{I}_{\pm\mathbf{x}}(\mathbf{s}) n_x^2 + \mathbf{I}_{\pm\mathbf{y}}(\mathbf{s}) n_y^2 + \mathbf{I}_{\pm\mathbf{z}}(\mathbf{s}) n_z^2.$$

Notice that our directional interpolation (related to [MFE07]) is of course exact when the normal $\mathbf{n}$ is aligned with $\boldsymbol{\delta}$. This interpolation basis is composed of the diagonal polynomials of the second order *radiation pressure tensors* [Arv95]. Discussions on more accurate approaches are presented in Appendix B.

3.2.2 Spatial Reconstruction

This previous definition is still tied to a position on a surface and such dependent on the geometric complexity. To get rid of this dependency, we thus compute a set of irradiance vectors at vertices of a 2D regular grid (i.e., a 2D texture) for quasi-planar surfaces and of a 3D regular grid (i.e., a 3D texture) for more complex objects. Such an approach is required for highly detailed scenes [CB04]. In the following, we consider mainly 3D grids, since in general the 2D adaptation is straightforward.

In order to compute smooth indirect illumination, we spatially interpolate the irradiance vector for each point $\mathbf{p} = (p_x, p_y, p_z)$, by performing either a trilinear or a tricubic (resp. bilinear or bicubic) spatial interpolation of the irradiance vectors stored at the 3D (resp. 2D) grid vertices $\mathbf{v}^{ijk}$ surrounding point $\mathbf{p}$:

$$\mathbf{I}_{\boldsymbol{\delta}}(\mathbf{p}) = \sum_{ijk} \lambda_{ijk}(\mathbf{p}) \mathbf{I}_{\boldsymbol{\delta}}^{ijk}.$$

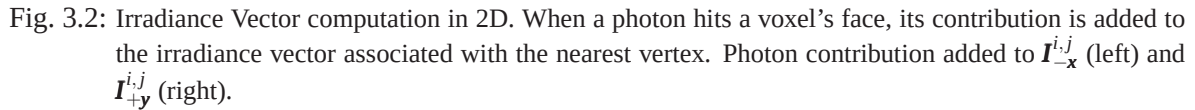
3.2.3 Precomputation

Any global illumination technique may be used to estimate the irradiance vectors stored in the grid.

Particle Tracing for 3D Grids

In our implementation, we use Particle Tracing by propagating photons from the light sources through the grid. Every time a photon traverses a voxel's face, its contribution is added to the irradiance vector $\mathbf{I}_{\boldsymbol{\delta}}^{ijk}$ associated with the nearest vertex $\mathbf{v}^{ijk}$ and the direction $\boldsymbol{\delta}$ provided by the normal of the face (see Figure 3.2). When a photon hits some scene geometry, a classical stochastic reflection is applied according to the local BRDF.

The photon propagation is accomplished in two steps. First, using a ray tracing acceleration structure [Gla89a], we find the closest intersection with the scene. Then, we propagate the photon into the Irradiance Vector Grid without any intersection test. Once all photons have been treated, a


$$I_{\delta}^{ijk} = \frac{1}{A_{\delta}^{ijk}} \sum_{n=1}^{N_{\delta}^{ijk}} \phi_n \omega_n h(\delta, \omega_n)$$

Note that our approach does not suffer from the classical boundary bias of photon mapping (where spheres with large radii are used to collect photons, such as in room corners and along contours of flat surfaces) as our density estimation correctly accounts for the intersection of the photons with the grid. Unlike the strategy of Havran et al. [HBHS05], our approach does not need to store all the rays generated from the photon propagation.

$$\mathbf{M}_{\delta} = \begin{bmatrix} \mathbf{I}_{R,\delta} & \mathbf{I}_{G,\delta} & \mathbf{I}_{B,\delta} \end{bmatrix} = \frac{1}{A_{\delta}^{ijk}} \sum_{n=1}^{N_{\delta}^{ijk}} \omega_n^T \phi_n h(\delta, \omega_n),$$

Path-Tracing

60

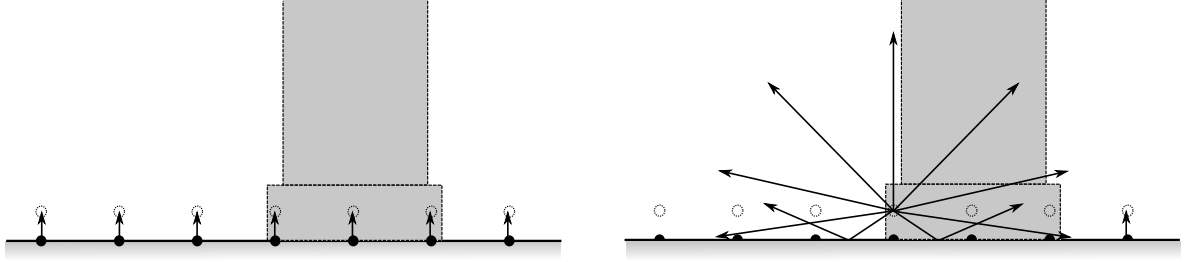


Fig. 3.3: Sampling strategy for 2D grids. (left) a small translation in the direction of surface normal is applied on vertices. (right) Because samples can now be inside other objects, no intersection is taken into account in the precomputation of indirect lighting before leaving them.

N rays are issued in directions ω_n and their corresponding radiances l_n ($W.m^{-2}.sr^{-1}$) are accumulated for each principal direction δ :

$$I_{\delta}^{ijk} = \frac{1}{N_{\delta}^{ijk}} \sum_{n=1}^{N_{\delta}^{ijk}} \frac{l_n}{p(\omega_n)} \omega_n h(\delta, \omega_n).$$

In this equation, $p(\omega_n)$ stands for the density of probability in emitting a ray in the direction ω_n . Similarly to the photon mapping case, the final irradiance vectors are colored ones and are expressed as the following matrices:

$$\mathbf{M}_{\delta} = \begin{bmatrix} I_{R,\delta} & I_{G,\delta} & I_{B,\delta} \end{bmatrix} = \frac{1}{N_{\delta}^{ijk}} \sum_{n=1}^{N_{\delta}^{ijk}} \omega_n \mathbf{l}_n \frac{1}{p(\omega_n)} h(\delta, \omega_n),$$

where $\mathbf{l}_n$ is the colored radiance of ray n .

For all 2D grids vertices, we precompute the irradiance vectors with a simple path tracer [DBB06]. Unfortunately, if the 3D position of a grid vertex lies on the associated surface, only half of the directions are accessible. Similarly to Arıkan et al. [AFO05], a small translation in the direction of the normal is applied (cf. Figure 3.3-left). Unfortunately, some displaced vertices can now be inside other objects and therefore unwanted shadowed areas may appear. To solve this problem, we ignore the intersections that occur inside these objects (cf. Figure 3.3-right).

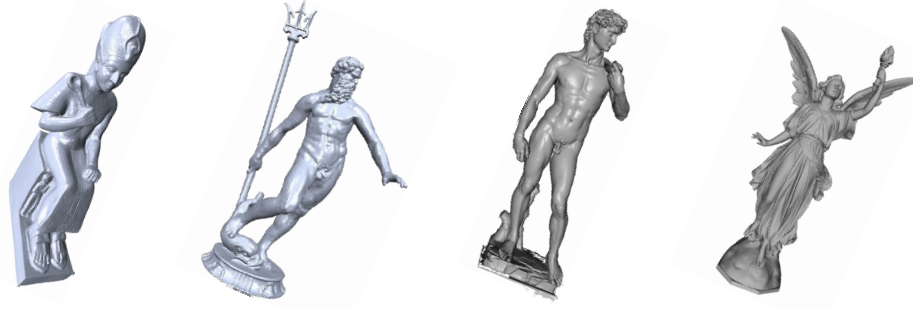
Note that, the two approaches, path tracing and particle tracing, can be easily parallelized. Note also that no intermediate structures are required for storing the impacts, leading to a small memory footprint for the precomputation.

3.2.4 Compression

Remember that Equation 3.3 defines an irradiance vector for a single color component. Since, in computer graphics chrominance is defined by three primary colors (R, G, B), we need, for each δ , three irradiance vectors stored in a 3×3 matrix $\mathbf{M}_{\delta} = [I_{R,\delta} \ I_{G,\delta} \ I_{B,\delta}]$. For a given grid vertex and a given δ , we compress $\mathbf{M}_{\delta}$ as the product of a direction $\mathbf{d}_{\delta}$ and a color $\mathbf{c}_{\delta}$ defined as follows:

$$\mathbf{M}_{\delta} = \mathbf{d}_{\delta}^T (\mathbf{c}_{\delta})$$

$$\mathbf{d}_{\delta} = \frac{I_{R,\delta} + I_{G,\delta} + I_{B,\delta}}{\|I_{R,\delta} + I_{G,\delta} + I_{B,\delta}\|}$$



	Ramses	Neptune	David	Lucy
Full / Simplified	1.6 M / 80 K	4 M / 100 K	7.2 M / 100 K	15 M / 150 K

Fig. 3.4: Size (in polygons) of the full and simplified objects.

and

$$\mathbf{c}_\delta = \frac{1}{\tau_{\mathbf{d}_\delta \delta}} \begin{bmatrix} \tau_{R,\delta} \delta & \tau_{G,\delta} \delta & \tau_{B,\delta} \delta \end{bmatrix}.$$

This guarantees that when the normal $\mathbf{n}$ is aligned with δ , we preserve the original RGB intensity: ${}^T\mathbf{M}_\delta \mathbf{n} = \mathbf{c}_\delta {}^T\mathbf{d}_\delta \mathbf{n}$ (using Equation 3.4). We have tested experimentally that this compression does not introduce any artifact in the indirect lighting interpolation. More accurate approximation might be achieved using an iterative optimization as shown in Appendix B - Section B.1.4.

To reduce the required bandwidth usage even further, the direction can be quantized on 24 bits (classical quantization with 8 bits for each coordinate) and the color on 32 bits, using the GPU-compatible R9_G9_B9_E5 format similar to the RGBE format [War91]. These two vectors are encoded in two 3D textures, and therefore the information for the six δ directions requires 12 3D textures. To reconstruct the indirect lighting, one may use trilinear interpolation natively provided by the hardware, or adapt a tricubic interpolation technique [SH05].

3.3 Applications and Results

As said in the introduction, the vector representation has been developed intended for an improved user-control on the indirect illumination. Nevertheless, we have to prove that this representation is first valuable to compute a global illumination solution. This illustrates that a user-intended development can also improve existing techniques. In this section, we thus present its applications into three different contexts.

3.3.1 Geometric Complexity: Precomputation and Hardware Rendering

More and more objects used in 3D graphics are issued from 3D scanning. This introduces the large richness of real world. This *original geometry* can be too large, to offer interesting performance when precomputing the global illumination. However, since indirect illumination is a slow varying function [WRC88], it does not depend on all the original details. A *simplified geometry* is thus largely sufficient and it will greatly speed up the preprocess as shown by Tabellion and Lamorlette [TL04]. Our geometry-independent representation is well-suited for such an approach. This coarse level can also be used for distant objects. However, for closer viewpoints, all tiny details can convey important visual information. These details are re-introduced in the intermediate level of details for interactive rendering.

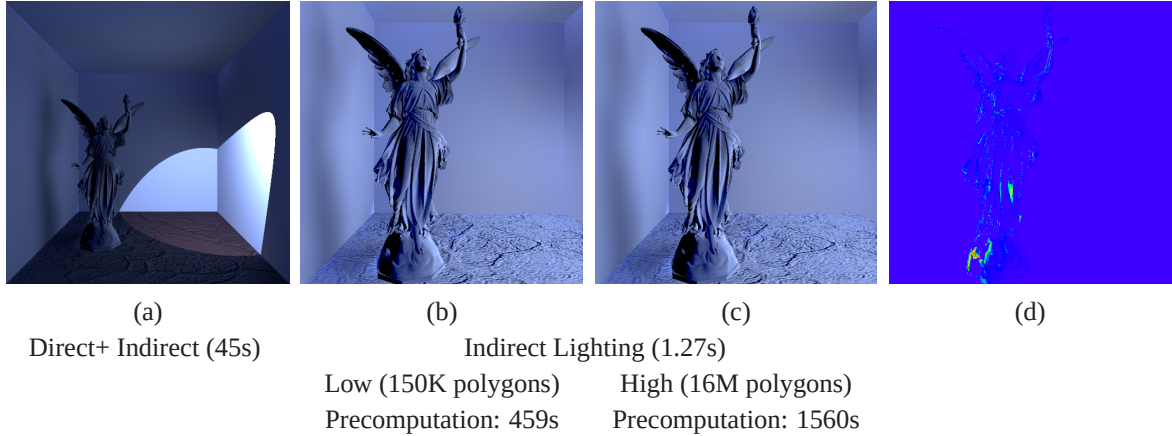


Fig. 3.5: Precomputation on the full versus simplified geometry. The resolution of the 3D grid used to capture the indirect illumination around this statue is $8 \times 16 \times 8$. In (b) and (c), we use software rendering to show the indirect irradiance from the grid precomputed on respectively low (b) and high (c) geometry. Computing the difference in *Lab* color space (d) between (b) and (c) shows that the maximum error is 69 (15% of maximum possible error) and is very localized. The average error is 0.4 (0.09%).

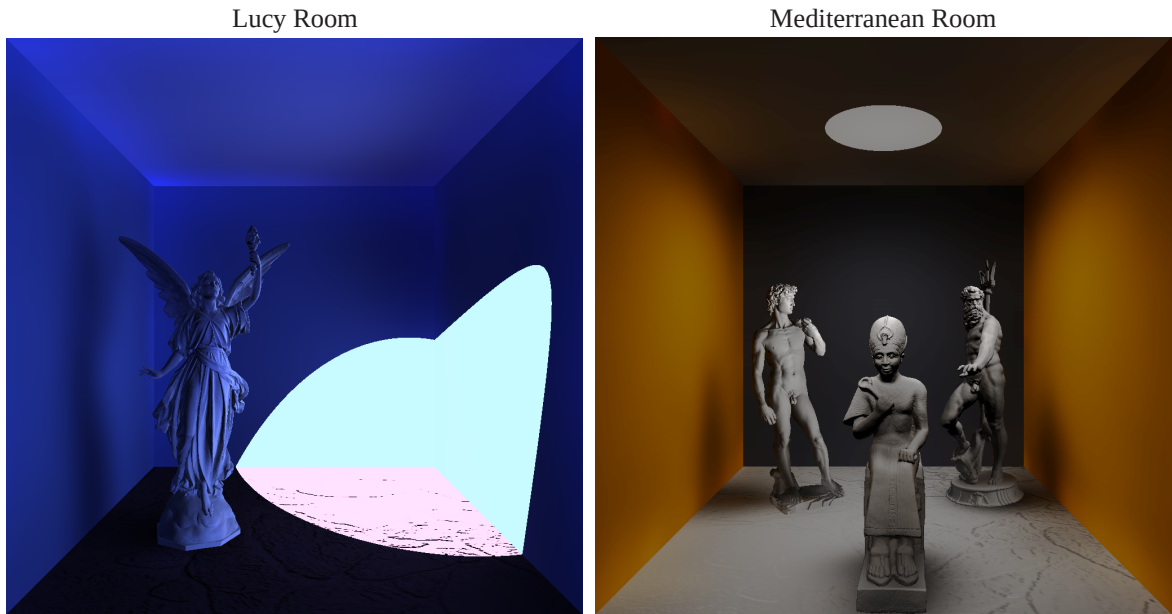


Fig. 3.6: The test scenes. The first scene is dedicated to Lucy statue. The floor consists in a normal mapped quad and a spot-light is directed toward the back-right corner. The second scene is dedicated to statues gathered from around the Mediterranean Sea from different places and periods. It consists in three statues, a normal-mapped floor and a spot light directed toward the ceiling.

This approach allows to speed up precomputation without losing too many features in the illumination. To validate this point, we compare (cf. Figure 3.5) the indirect irradiance obtained when using grids precomputed with the simplified (cf. Figure 3.5(b)) and full geometries (cf. Figure 3.5(c)). As shown in Figure 3.5(d) the average difference in perceptually uniform *Lab* color space is very low. The maximum possible error reaches 15%, but mostly in darker areas where it is less noticeable.

Most of the original details can be represented by a normal map with sufficient visual accuracy. We use *simplified meshes with normal maps* for the final rendering step. This representation enables

	Lucy Room	Mediterranean Room
Full / Simplified geometry	15 M / 150 K	12.8 M / 280 K
APO	21 MB	94 MB
3D grids	1 / $8 \times 16 \times 8$ / 144 KB	3 / $16 \times 16 \times 16$ / 1.7 MB
2D grids	6 / 32×32 / 264 KB	6 / 32×32 / 264 KB
Lighting Precomputation	55 min	260 min
Full geometry	3-8 fps	5-8 fps
APO	35 fps	30 fps

Tab. 3.1: Size of the different structures and average rendering frame-rate. The geometry size corresponds to the number of polygons. We present in parentheses the number of 2D and 3D grids used in each scene, their resolution, and the total amount of memory. Note how small the latter is compared to the size of the simplified geometry. We use 16K paths for each grid vertex. The frame-rate has been evaluated for a 800×800 pixels viewport. For a given scene, the same grids are used for indirect illumination with the different representation of the geometry.

real-time performance while preserving the visual richness of an object. Since global parameterization of a large object can be difficult to compute, we rely on a volumetric and parameterization-free representation: the appearance-preserving octree (APO) [LBJS07]. An APO encodes a normal map in an octree texture.

We experiment our grids structure for indirect illumination with two scenes (cf. Figure 3.6) where the illumination on the main objects is thus mostly indirect. In all scenes, we use 2D grids to capture the indirect illumination and shadows for the walls, floor, and ceiling, while the 3D grids capture them for the archeological objects. For each scene and as described in previous paragraphs, we use a simplified mesh to precompute the indirect illumination, and this mesh with normal mapping for the final interactive rendering. Thanks to our approach, our system allows users to navigate in 3D scenes and visualize the objects interactively whereas this would be unattainable with full geometries (cf. Table 3.1). Note also that the overhead due to our representation of the indirection illumination is low compare to the required memory for the geometry (cf. Table 3.1).

3.3.2 Streaming

We have also experimented our representation on a client/server architecture. The server precomputes and stores the lighting structures and the levels of detail (LODs) for each complex geometry represented as a progressive mesh. The server sends either new geometric or lighting level of detail depending on the client requests. After each data reception, the client performs some processes on the illumination structure and on the progressive mesh before uploading them on the GPU. Moreover, our approach allows to interleave geometric and lighting data when transmitting the scene from the server to the client. This offers a very smooth progressive visualization until the desired quality is reached.

Streaming of Irradiance Vector Grid

A classical solution for the progressive transfer of the texture is to use a hierarchical decomposition based on recursive basis functions such as wavelets. However, to reconstruct each hierarchical level, this decomposition technique requires waiting for the reception of all corresponding detail coefficients. Thus, the time required to obtain each new resolution level is growing with its size. We propose here an alternative approach, which allows the transmission of constant size bundles of voxels.

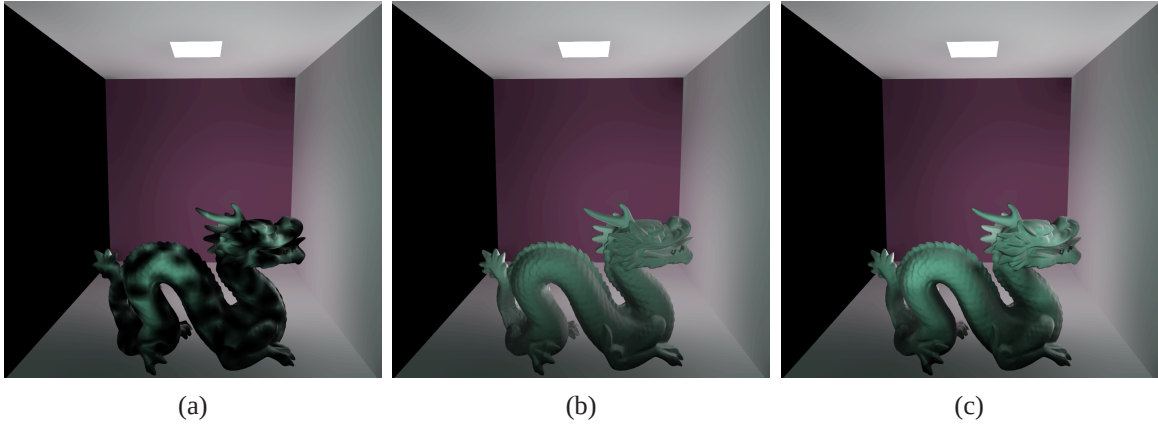


Fig. 3.7: Comparison of indirect illumination reconstructed on the right side of the dragon (a) without our push-pull algorithm, (b) with and (c) with a push-pull algorithm without smoothing. The grid dimension is $16 \times 16 \times 16$ and half of the vertices have been transferred. The push-pull process has filled the darker regions of the dragon with smooth indirect illumination.

The transmission is initialized by transferring the eight corners of the grid. Then, each client request consists of a constant number of irradiance vectors. Notice that the number of irradiance vectors per request can be dynamically set depending on the client GPU/CPU capabilities as well as the network bandwidth and reliability. To get a smooth global update of the indirect illumination, we have implemented a stratified random sampling of the grid. In our current implementation, the grid is divided in a set of slices along its longest axis. Then, at each client request, the server sends the requested number of irradiance vectors. The locations of the irradiance vectors are randomly distributed on each slice.

Unfortunately, when streaming the IVG, some illumination holes may appear until the grid has been fully transferred. This comes from the fact that the incomplete grid holds invalid irradiance vectors at some locations. We have adapted a 3D push-pull algorithm to fill the missing irradiance vectors with smooth data interpolation (cf. Figure 3.7). Notice that our push-pull algorithm does not modify any received data. To get smoother results, we apply after each push step a pyramidal filter to the current grid level. Finally, the push-pull process can be skipped if the client has limited CPU capabilities. Therefore, the minimal hardware requirement for our client is programmable graphics hardware with 3D texture support.

The direction and color of each irradiance vector (cf. Section 3.2.4) is sent either in floating point format or quantized. Our experimentations have shown that using quantization reduces the transfer time by a factor of about 2.5 without introducing any visible artefact. Indeed, for the scene presented in Figure 3.8, the mean difference between an image reconstructed with and without quantization is only 0.008% in *Lab* color space. Obviously, the dequantization process must be performed on the client side in order to perform the push-pull steps, but the resulting overhead is very small (cf. the chart of Figure 3.9).

Results

We have tested our remote 3D visualization system with an Intel Q6600 with 4GB memory as a server and an Intel Pentium M 2.26Ghz with 2GB memory as a client. We have measured all network transmission times on a 802.11g WiFi network. Each measurement has been repeated and averaged.

One main advantage of our separation between the illumination structure and the geometry struc-

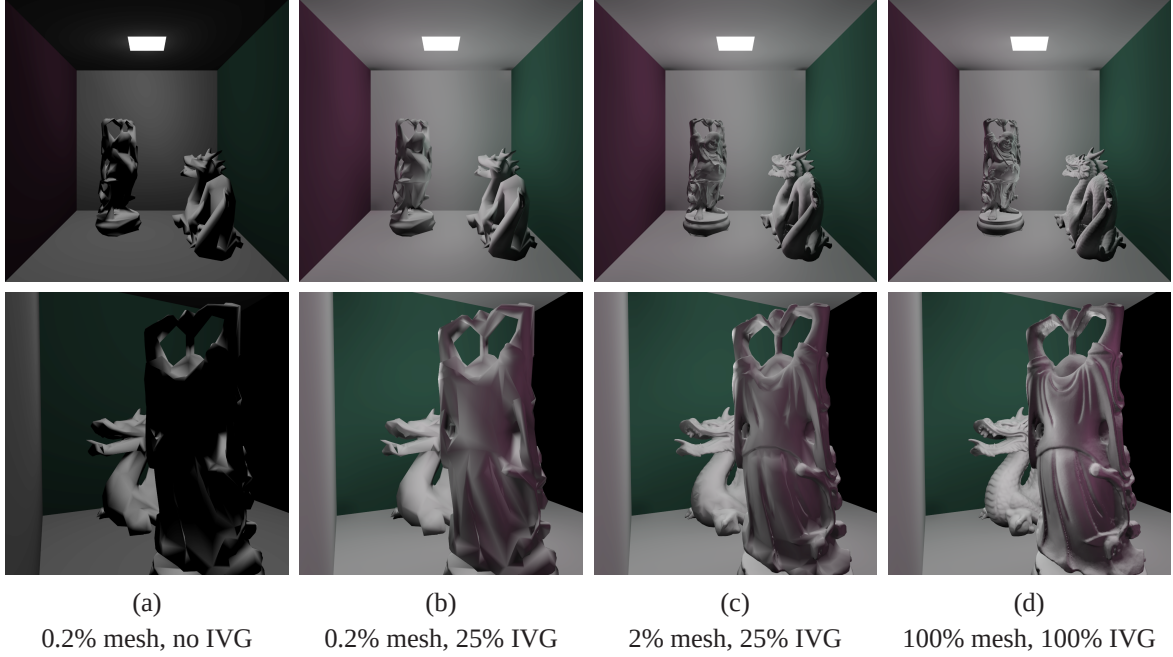


Fig. 3.8: Our client/server visualization system to stream alternatively geometry and lighting. (a) A scene with 0.2% of the geometry transferred and only direct illumination (without shadows). (b) The same amount of geometry with 25% indirect illumination transferred: color bleeding effects are now included, like on the surface of the Buddha oriented toward the red wall that appears redder. (c) Further refinement of the geometry (2%). (d) Full-resolution geometry (50 MB) and full-resolution (1 MB) irradiance vector grid. This scene runs at 50fps on a NVIDIA GeForce Go 7800 GTX.

ture is that the client adapts its data refinement demands according to both hardware and network capabilities. The classical strategy that offers the smoothest progressive visualization is to use an interleave streaming of illumination and of geometry (cf. Figure 3.8). Our tests show that the client frame-rate remains constant, when updating either the geometry or the illumination grid on the GPU. Therefore our system provides a real-time and continuous feedback to the client. Moreover, for a given scene, we did not measure any frame-rate performance penalty when introducing the illumination grid.

We have tested both the streaming of quantized and floating point grids. As expected, the network transfer time is reduced when using a quantized grid (cf. black curve on the two graphs of Figure 3.9). Moreover, the comparison of the two red curves of Figure 3.9 demonstrates that the data dequantization process is very small (approximately 4%). This dequantization step is required to perform the push-pull algorithm without introducing numerical errors. The time spent on the push-pull algorithm is constant per grid size, but the number of push-pull processes depends on the number of client requests. Therefore, the packet size has to be chosen carefully depending on client/server and network capacities. According to Figure 3.9, when using a $32 \times 32 \times 32$ quantized illumination grid, the appropriate packet size is reached when asking approximately 100 samples per slice (the corresponding abscissa when the black curve crosses the blue curve). Finally, as illustrated in Figure 3.10, most of the computation overhead is due to the smoothing pass of the 3D push-pull algorithm. However, depending on client capabilities, this step can be skipped and as shown in Figure 3.7, the penalty on quality remains small.

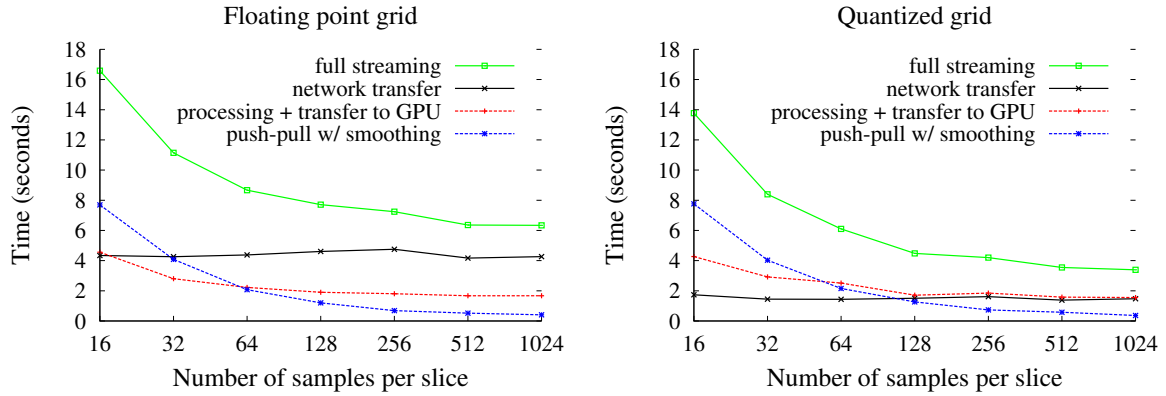


Fig. 3.9: Downloading time for the complete illumination grid ($32 \times 32 \times 32$) with different buffer sizes. The size of the floating point grid is 3 MB and the size of the quantized grid is 1.3 MB. The processing includes the dequantization process and the copy to CPU memory.

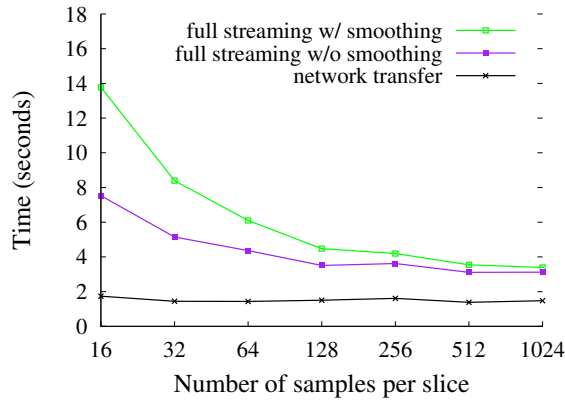


Fig. 3.10: Comparison of downloading time with and without smoothing during the push-pull step for a quantized $32 \times 32 \times 32$ grid. The smoothing represents 80% of the total time spent for the push-pull process.

3.3.3 Caching

Our Irradiance Vertex Grid can improve software global illumination. As it provides a smooth spatial and directional reconstruction of irradiance vectors, it can be directly used for diffuse indirect illumination, without propagating secondary rays for final gather. Direct use of cached values could be done using existing techniques [WRC88, KBPŽ06] but they do not guarantee a continuous reconstruction of the indirect illumination. Moreover, as they do not offer geometry robustness, these techniques require a large number of samples for highly-detailed geometry, in order to provide accurate estimation.

A final usage of our Irradiance Vector Grid is to be employed as an efficient caching structure for stochastic approaches. For instance, as done with photon mapping, our grid may be accessed indirectly by shooting secondary rays from points being shaded. The main advantage is that high-frequency details such as indirect soft shadows are well preserved, and that reconstruction errors are masked since a diffuse or low-glossy reflection is similar to a low-pass filter [DHS⁺05]. So, a simple trilinear spatial interpolation provides a good compromise between speed and quality. All the following results have been computed on an AMD 64bit 3500+ processor with 2GB of memory.

To evaluate our approach, we present two test scenes with complex lighting. The first configuration (Figure 3.11) presents a scene mainly directly illuminated from 11 light sources. The second



Precomputation: 201s
Rendering: 1,341s (51s)

Precomputation: 252s
Rendering: 6,618s (5,527s)

Fig. 3.11: Scene with mostly direct illumination case. 16 rays per pixel were used. 5 M photons shot. (left) Our technique with a $40 \times 50 \times 40$ uncompressed grid (24.5 MB) used directly with a tricubic interpolation scheme. (Center) Photon mapping with Christensen’s Cache using 50 photons of the 5 M (124 MB) stored where used to precompute each irradiance sample. (right) Reference solution obtained by Path-Tracing with 1600 rays per pixel. The overall illumination is similar in all three techniques, however our much faster indirect illumination computation (51s vs. 5527s) strongly reduces the total rendering time (1341s vs. 6618s).

one (Figure 3.12) is a classical two-room configuration where one room is indirectly illuminated by a light source placed in the other room. Both scenes have more than 8 million polygons due to the highly detailed objects. They illustrate the geometric robustness of our approach.

We compare our technique to photon mapping combined with Christensen’s [Chr99] precomputed samples. Precomputation time for our technique involves shooting photons and accumulating their contribution in the grid, and for Christensen’s it involves shooting photons, balancing the kd-tree, and precomputing irradiance. For all images generated with Christensen’s method, we set the number of precomputed samples to be one fourth of the total number of photon hits, as originally suggested [Chr99].

Figure 3.11 compares the results obtained with our technique and Christensen’s for direct lighting configuration. For fair comparison, a reference solution has been computed with a high density (1600 rays per pixel) Path-Tracing algorithm. For equivalent precomputation times, our technique allows a much faster computation (51s vs. 5527s) of the indirect component of the illumination due to direct access to the cached values. In our technique, the final gathering step spends most of its time computing ray-geometry intersections and direct illumination, while Christensen’s cache requires casting a large number of additional gathering rays, since direct access produces very objectionable illumination patterns with zones of constant irradiance similar to Voronoi diagrams. This is a well-known artefact [Chr99] preventing direct access to cached irradiance.

For the indirect lighting configuration presented in Figure 3.12, we perform a direct access to our structure but to capture finer shadows cast by the columns, a higher-resolution grid ($30 \times 20 \times 92$) was built and consequently a larger number of photons needed to be shot. This resulted in a much larger precomputation time but our method with direct access is still much faster (2,311s vs. 47,971s) than Christensen’s caching technique for a similar quality.

Alternatively, we have also tested our technique with an indirect access (cf. Figure 3.12) to the cached values by using a lower resolution grid ($12 \times 8 \times 20$) traversed by the same number of photons as with Christensen’s method. Our technique reduces both precomputation time (150s vs. 466s) and reconstruction time of the indirect illumination (2,376s vs. 10,050s), while retaining similar illumination features. Access time to our grid is constant in the number of the cached samples instead

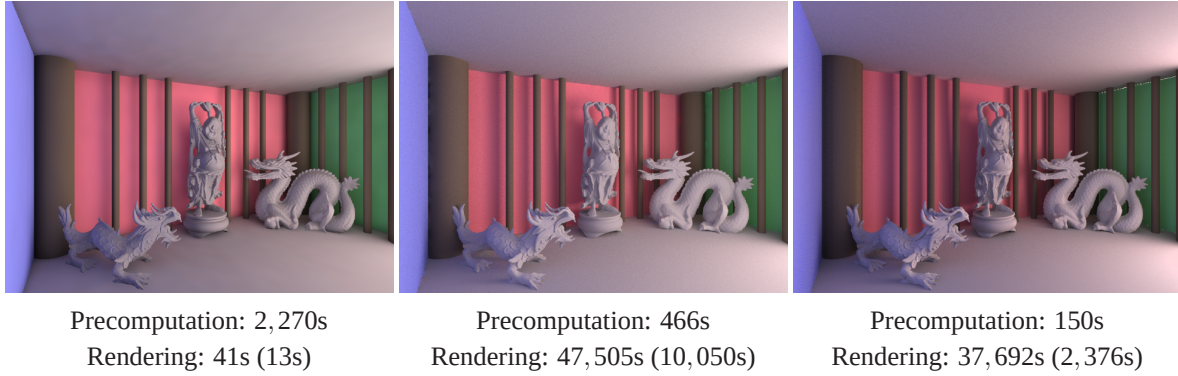


Fig. 3.12: Scene with mostly indirect illumination. (left) Our technique using directly a $30 \times 20 \times 92$ uncompressed grid (11.4 MB) constructed with 80 M photons. (Center) Christensen’s method reference image with 3200 rays to sample the hemisphere and 5 M photons stored (124 MB). (right) Our technique using indirectly a $12 \times 8 \times 20$ uncompressed grid (590 KB) with the same number of photons shot and the same number of rays to sample the hemisphere.

of being logarithmic as with a kd-tree [Chr99, CB04] which explains these gains.

3.4 Conclusion

In this chapter, we have introduced a new representation for indirect illumination. Based on geometry-independent grids (2D for quasi-planar surfaces and 3D for generic surfaces) and vector irradiance, this approach is well suited for rendering complex scenes with global illumination. Our structure is robust to simplification techniques with appearance preservation. To demonstrate this point, we have integrated our structure with geometry streaming techniques in a remote visualization system to quickly provide global illumination effects for complex geometries. Furthermore, the transfer time overhead induced by our structure is very small as well as the overhead on the performance at rendering time.

We have also implemented this structure both as a 3D texture for indirect illumination on the GPU and as a caching scheme for Photon Mapping. The results show that diffuse inter-reflections are well captured. Compared to existing solutions, our approach requires fewer cached samples for higher-quality cached indirect lighting. Additionally, it does not require any photon storage. Furthermore, our irradiance cache can be directly accessed during the final gathering pass.

Based on these first validations, we have to accelerate the precomputation of our structure, since we do not take currently into account the high coherency of the sample paths. This development has also to take into account the generalization of parallel architectures (clusters, massively multi-core processors such as GPU [SCS⁺08]). Such improvements would result in more interactive modification in 3D scenes, required for user-edition.

Furthermore, more robust directional basis for vector representations (see some investigations in Appendix B) would improve the interpolation scheme and provide a more accurate solution. Since the current scheme is based on linear interpolation, some artefacts may appear for complex variations of illumination. The introduction of gradients might also improve the smoothness of the reconstruction where incoming illumination varies quickly.

And finally, the irradiance vectors are currently evaluated on regular grids. Identifying proper resolutions for the grid and the number of photons to shoot requires a good understanding of the illumination effects. A more automatic estimate would facilitate the integration of this technique. For

large scenes, multiresolution and adaptive structures would reduce the construction cost. This will also allow better capture of indirect illumination near surfaces.

In this chapter, we have presented a limited set of applications and estimations methods for our vector representation. I believe that such an approach could be extended to a larger range of applications, like radiosity methods, ambient occlusion (cf. Appendix B), more glossy and specular reflections and vector reflectance. For such extension, I believe that a vector representation could improve the visual accuracy, by reducing the dependency toward the geometry.

But the most important extension is the finalization of this work for user-edition. Intuitively, this representation is similar to a field of directional light sources that we are used to manipulate in Computer Graphics [TL04]. This was our original goal that has still to be realized. The idea would be to locally adjust the colored intensity and direction of an irradiance vector, and to propagate this modification in the grid by minimizing the change with original solutions. This minimization process might help in preserving the global coherency of the illumination.

Conclusion & Future Research

Along this document, we have explored the different steps that are usually required for image synthesis, from modeling to rendering. The main focus of this exploration was to take as much as possible users into account during the development of the underlying representations. Indeed, the main difference between Computer Graphics and simulation, as said in the introduction, is the creative aspect: final images have to be as close as possible to user desires. In this conclusion, I summary the resulting contributions and introduced the general approach for my future researches.

4.1 Contributions

The creation of 3D images relies first on the definition of a collection of 3D models and on their relative positioning. For this purpose we have proposed a new **sketch-based modeling tool** for 3D meshes extending the use of shape curves from silhouettes to profile. To ease the use of such modeling tools, we have exposed intuitively all the different steps of the reconstruction process together with the underlying representation, leading to a multi-view interface. We have also explored the use of such modeling tools when users are standing in front of large displays. From these contributions, one main conclusion is that one way of creating intuitive tools is to create hybrid solutions using 2D interactions for resulting 3D objects.

Once the object geometry is defined, users have to work on designing its appearance which is the result of light interaction with the underlying reflective material and geometry. We have thus introduced a new **BRDF model and its modeling tools** to extend the range of possible lighting effects. This model is defined to allow real-time sketching and painting metaphors for BRDF design. A lobe of our BRDF is represented with a polar curve for its shape and a texture for its color gradient and for precise shape enhancement. Since the object shape participate also the shading perception [VLD07], we have also introduced a **shape descriptor** to extract continuous information of convexity and curvedness based on a combination of object-space and image-space techniques. This real-time analysis is used to re-introduce them in different shading styles.

Finally, we have introduced a new lighting representation that may potentially be used for user edition of global illumination. This representation is based on a **regular grid of vector quantities** storing the incoming lighting. The combination of volumetric and vector representation make it suitable for all algorithms that have to deal with the geometric complexity of a 3D scene. We have experimented

it for GPU rendering, 3D streaming with illumination and illumination caching.

All these contributions lead to some further researches for both improvements of the presented techniques and for their extensions to other applicative domains. Their specific limitations have been discussed in the conclusions of the related chapters. Despite the main common limitation that is, the lack of user study to evaluate the user efficiency in a creative context, we believe that the presented research approach provide and will provide useful materials for further investigations and validations in cognition and perception. These materials consist in a collection of tools and visual effects that can be use to design new experimentations. Despite this potential, such investigations are complex and long to perform by scientists from Computer Graphics: they require skillful specialists in cognition and perception, and in the design of such analysis. Nevertheless, we need reciprocally to take more into account the results issued from these scientific domains, in order to enhance the legibility of computed images, or to save some computation. Collaborations with these domains would thus be more based on exchanges than on common work.

4.2 Future Research: General Approach

All the results presented in this document are issued from a same approach: the first step of any new development is to precisely define the controls that could be provided to users, even if the resulting representations are potentially used in a wider range of contexts. We will continue to explore the solutions that could emerge from this approach, taking into account that nowadays, techniques issued from Computer Graphics are used in a large variety of domains. For a personal point of view, I will mainly focus on *rendering models* and corresponding *acquisition systems*.

4.2.1 User-centric Computer Graphics: Definition of Contexts

Who are the users? Due to this large variety of domains, the generic term of users hide in fact an amazing variety of persons and applicative contexts. In the current document, artists [Sei99] were considered as the most representative and specific users for Computer Graphics: they are characterized by a need of freedom to be creative and their skills due to generally long training in using the required tools. But a lot of works in Computer Graphics focus more on scientific users: they are characterized by a large requirement in quantitative accuracy that leads to the development of faithful simulations like in global illumination. With the increasing presence of computer everywhere, the focus can not be anymore these only assumptions and more diversified users have to be considered: their skills are different depending their habits and trainings, common users often need simplified but more constraining tools to start any process in order to reach some average correct results; some specific domains require the introduction of new expectations, skills and experience that characterize their users.

What users want to see? Since the main aim of Computer Graphics is in the production of images to be seen, this is the first question to ask in order to precisely define what stand for this term of "users". Depending on the context, users would probably want to see some as precise as possible results for a simulation, or some only realistic-looking solutions when the precision is not required, or fully detailed images independently on the rendering style. When it comes to application like scientific visualization, the goal is transferred from realism to legibility and expressivity: the resulting images could guide users to explore and visually extract some specific information of 3D world.

What users want to do? To extend the definition of who are the users, this question has to be also asked. Since Computer Graphics is becoming present in a large part of our everyday life, the goals of the developed applications have to be clearly defined. The development of a unique approach is quasi-impossible, and each solution has to be adapted to the specificities of each context. For scientific applications, the goals could be to compute a simulation, to create a demonstration, to explore or to illustrate some results. The main user concern is the precision, and the possibility to adjust it. Some users could be concerned by an efficient communication of visual information, like for education and teaching, for public presentation of scientific results. A large part of potential users are still interested in the gaming possibility offered by Computer Graphics: they are mainly concern by visually pleasant images and/or interactivity. For more specialized and trained users, they might wish to be creative: the main goal is to obtain the desired result and thus, reducing the freedom by too strong physical constraints can lead to the impossibility to reach this result.

Hybrid 2D/3D approaches. Once users with their goals, expectations and skills are well defined, they can be integrated into a Computer Graphics processing loop. To this purpose, interactions with the future algorithms have to be defined before. They will thus be based on hybrid approaches: the final results have to be 3D, and since 2D seems for easier interaction since I believe that they are generally more convenient due to the required support (like a tablet, touch-screen, surface, ...) and that accurate positioning in 3D rely on accurate visualization system that are difficult to build.

4.2.2 Getting away from Physics

In creative and visualization contexts, easy controls and legible rendering are more important than accurate physical simulations. Even more, a perfectly accurate and physical correct result without any control can be creatively frustrating. To achieve these goals, the physical constraints have to be weakened to increase users' freedom. As an example, I more and more do not believe that BRDF is the appropriate representation of lighting interaction for Computer Graphics: its underlying physical assumptions introduce too many constraints even for some realistic renderings.

This freedom does not mean the complete lack of constraints in developments for Computer Graphics. The main one is the definition of efficient controls before any new development. These controls have to take advantage of the knowledges and abilities of specific users, and more important, the effects of each control have to be easily identifiable: when it is possible, similar parameters with contradictory effects have to be prohibited.

This freedom does not always mean the complete lack of realism. Depending on the context, controls have to help users in creating plausible behaviors. A good trade-off between freedom, context and control has to be thus defined. I believe that plausibility might be preserved in editing existing materials and models: it might be extracted from the original representation and reported in the modified one by minimizing the changes due to users' edition. Another approach is to precisely define what the meaning of plausible behaviors is, and to integrate these definitions in the development of specific models.

To help users in understanding the controls and underlying processes, legible visual feedbacks have to be used to communicate some internal information. Expressive rendering can also help in enhancing the presentation to users of some characteristic features of 3D world and results in order to guide and help their exploration. Such a rendering can also be used to emphasize some non-quantitative information in-between 3D models.

4.2.3 Falling back into Physics

Interestingly, this move away from physics and the required definition of pertinent and understandable controls leads to and requires a better understanding of the phenomena themselves. This better understanding and the resulting representations also result in the development of more efficient algorithms for simulation, as illustrated in Section 3. This remark can be extended to Computer Graphics in general: this recent applicative domain relies on a large range of different sciences from traditional ones such as mathematics, physics, optics ... to human sciences such as cognitive science. For and from all these sciences, it creates a new context for classical simulation problems, resulting in new problems, approaches and solutions. The experience and knowledge gained in defining user-friendly algorithms can thus be reintroduced back into simulations.

More important accuracy is one of the possible desire and required control for some users. In such context, the physical constraint cannot be weakened to achieve a full simulation of some phenomena. More generally, when interaction with the real world or realism is a requirement of a given context (like for augmented reality or acquisition in general), physics is still required, mostly for the integration or the transformation of the real world into virtual images. This leads to the current development of a new trend in Computer Graphics, called Computational Optics.

Form a user point of view, Computational Optics acts as an interface between real and virtual worlds. Using optical systems makes possible to bring reality into virtual world and to project virtual word on the real one. Due to their interactions with real world, physical accuracy is one of the main concerns of such systems.

4.2.4 Personal Involvements

Over all the possible challenges in computer graphics and after my personal exploration of the different image synthesis steps, I will refocus my research on two topics: rendering models and acquisition systems.

Since the early days of Computer Graphics, rendering techniques were focused on physical simulation. But as shown in this document and in Section 4.2.2, the major requirement is to provide users with as intuitive and expressive as possible tools, based on their specific abilities. Not that intuitive do not mean that the tools have to be immediately understandable, but that their effects have to produce identifiable actions. Our works on shape depiction through shading [VBGS08a, VPB⁺09] have to be extended to more users' controls, explored in different applicative contexts and improved both in term of accuracy on analysis and in perception of shape and materials. Our researches on BRDF models have shown that getting away from pure physical simulation can still lead to plausible behaviors: new lighting representations and corresponding controls have to be developed. More generally, a large amount of research still need to be performed in order to reach full control on the different and complex aspects of shading (e.g., local vs. global, indirect vs. direct, shadows, level of expressivity ...).

The most important challenge is the second one: acquisition systems. Since one of the historical goal of Computer Graphics was to provide realistic images, and since the modeling and computational efforts have shown their limits in expressing all the amazing complexity of real worlds, we rely more and more on models issues from our understanding of reality, and in acquisition to bring this reality into virtuality, and how to combine them. As said in the previous Section, the main challenge is to develop some systems that act as interface between our world and our virtual models. These systems have to be adapted to the different requirements of different users, leading to the development of specific models and physical devices for each of the resulting contexts: the definition of expected

quality and the use of acquired models could vary over these different requirements.

4.3 Specific Context: Cultural Heritage

Over all the possible contexts, one has particularly raised my interest: Cultural Heritage. Surprisingly, the highly recent and fast evolving domain of Computer Graphics and the needs of cultural heritage that are, the need of preservation of the past, the need of propagation of this heritage and the need of understanding our heritage, both of them converge toward some common research problems. Interestingly, the users' needs in Cultural Heritage are very different and mostly span in the whole range of possibilities discussed in the previous section.

Accurate Modeling for Preservation and Evaluation: One of the first concerns in Cultural Heritage is its preservation to transmit our history to the next generations. Due to the increasing human pressure on environment and the natural degradation of relics, real preservation is not always a possible choice. In this context, numeric preservation could be one solution if the acquisition techniques and the required processing are sufficient accurate to be trusted. Note that database management has also to be taken into account, but the required knowledge is out of the scope of Computer Graphics. Moreover, Computer Graphics scientists can help in providing accurate simulations to reconstruct the past, and to thus evaluate different hypotheses.

Efficient Rendering for Presentation: Cultural Heritage belongs to humankind and cannot be only archived: it needs to be shared and transmitted. Computer Graphics can naturally participate to this diffusion in two contexts. The first one deal with the exchanges between scientists: illustrations, animations, and 3D models have to participate into the elaboration or the demonstration of certain hypotheses. To extend the impact of Cultural Heritage, most of the results have also to be publicly presented and used in, classically, museums and in leisure industry like tourism and Serious Games. This second context requires interactivity for which the accuracy constraints are often weakened to realism and replaced by fast feedbacks to users.

Interaction and Expressive Rendering for Exploration: Users issued from Cultural Heritage need to explore the archived data and different hypotheses for reconstructing the past. To explore this space, they need to be able to visualize concurrently different contextual information: an example, to understand the history of an ancient city, they may want to visualize of course its 3D reconstruction, but also the function and age of each building. They may also need to be able to visualize differently certain and uncertain reconstructions. All these manipulations for non Computer Graphics experts lead to the development of new interaction tools and visualization style such as Expressive Rendering.

Global Control on Colored Reflection

Color representations of illuminants (e.g., light sources) and reflectance are widely used in computer graphics. Since they have been designed to fit the color perception of our visual system, they have multiple advantages. First, they are the most compact representation for storing an image. Then, from digital camera to display, every system is color-based (mostly RGB). Secondly, they are easy to select or design.

As a consequence, these color representations are also used for the reflection computation, done by a component-wise multiplication. But unfortunately, since these color are only a projection of real-world spectra, and since the resulting color component are not orthogonal (see Figure A.1), this results in a coarse approximation. The most visible effect, is the lost of color when the illuminant and the reflectance are very different color (e.g., red light-source time blue reflectance). One can expect to have some transfers from one component to another during the reflection in order to still preserve the overall appearance.

Many solutions have been introduced to compensate for the restriction of the current trichromatic color-based reflection model: new color bases [Pee91], color transformations [FDF94, DF00, DF02, WEV02], spectral estimation from color [Gla89b, SFCD99, Smi99]. The most accurate solution is to process the full rendering pipeline with a spectral representation of light [RF91, IP00, ZCB98], but suffer from the data availability. All these solutions induce a large change of the current rendering process, and also modify the reflection behavior under a standard color illuminant. Moreover, they do not allow any control the global behavior of the reflection. We thus introduce a new color operator [GD04a] in this chapter as a first step toward this achievement.

A.1 Motivations

Our main goal is to provide some controls on the global behavior of reflections with a smooth transition from the current model. With such approach, users can do a smooth transition from standard reflection model to the new one. With these new controls users can modify the lighting of the scene at three levels: (i) designing the material properties of the surfaces, (ii) designing the light sources, and now, (iii) globally controlling how much reflections preserve color appearance of light and materials. We introduce only this last control. Note that, this third control level is even more important in global illumination since the resulting color of a surface depends on multiple bounces of light.

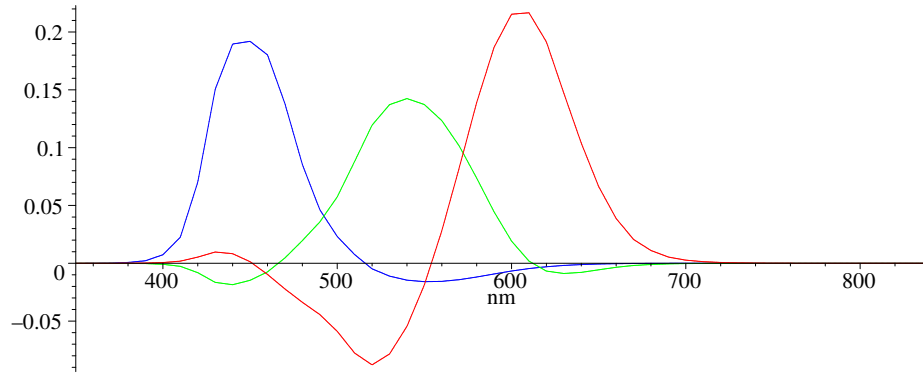


Fig. A.1: The three color matching functions used to project spectra into the ISO RGB color space.

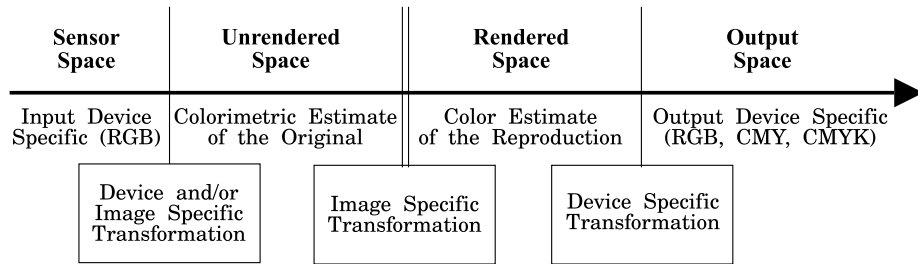


Fig. A.2: Digital Image Color Flow[SBS99]. In the first and last space, the color definitions are specific to the devices and to the manufacturers. In the *Sensor Space*, images are acquired and the color representations depend on the sensor characteristics. In the *Output Space*, images are finally rendered (e.g. on a RGB display or a CMY/CMYK printer). In the *Rendered Space*, the colors are transformed in order to be as close as possible to the ones available on the output device (e.g., tone-mapping). In the *Unrendered Space*, the color still contains its high-dynamic range information and can thus be easily transformed.

Our second goal is to be completely compatible with the current approach, that is,

- we want to keep the entire process in a unique trichromatic color representation, use along all the rendering process. For efficiency reason, we want to prevent un-needed transformations between color spaces. Here, we assume that the input data of most of the rendering systems is, and will be, color-based (images from cameras, scanners, etc.), similar to the output devices (considering LCD/CRT displays).
- We want to provide the same results for *constant* colored illuminants or materials (i.e., the illuminant color is (k, k, k) with $k \neq 0$). With such condition, users keep the exactly same approach to design light sources and surface properties independently, with the guaranty of a similar behavior under usual color illuminants (i.e., close to *constant*).
- We want to provide smooth transition from current reflection model to the new one.
- We want the new approach to introduce only minimal changes in current rendering systems.

Note that, with this model, we do not try to bring the richness and the accuracy of a full spectral rendering since the spectral definition of surface and light sources is rarely provided. Our goal is to reach a better behavior of reflections that is, preserving the color appearance, and preventing unwanted color disappearance - like for example green surface reflecting red light. Our goal is not to provide a transformation to spectra, for spectral rendering, since it will break the simple integration to existing rendering process.

Contributions

We propose to only change the operators describing the interaction of light and surface reflectance and to keep them commutative. It is desirable to introduce minimal changes to existing rendering systems. These operators span in the Unrendered Color Space (see Figure A.2), where the reflections are processed.

A.2 Previous Work

Most rendering systems use an RGB¹ color space as their Unrendered Space, since the input data - acquired by a camera or any other imaging device, or defined by a user - is usually only RGB color information, since the visualization output is in general an RGB display. Keeping the computation in this color space prevents to perform color space conversions.

Unfortunately, current rendering systems evaluate the light reflection on a surface by a component-wise multiplication. This fast scalar reflection is correct only for a constant illuminant, and for colors defined by normalized color matching functions, as shown in [Bor91]. As a result, some color information can be lost. For example, a blue wall can appear black under a yellow illuminant. This can be particularly apparent in global illumination algorithms since they take into account multiple reflections of light.

This issue is due to the fact that the color matching functions are overlapping and that they are not orthogonal. With the "Spectral Sharpening" approach [FDF94, DF00, DF02], it is possible to find the "sharpest" functions obtained by a linear combination of the original color matching functions. In this basis, the component-wise multiplication for the reflection is valid for a larger color range.

A commonly chosen solution to improve the reflection behavior is to use an appropriate spectral representation of lights and surface materials. Typically, a fine sampling of the spectral distribution is used. With their error-bound approach, Zeghers et al. [ZCB98] have shown that, in a diffuse environment, a global illumination result can be achieved with only four well-selected spectral samples without any visual difference. Such spectral approaches provide very precise results, including effects like color dispersion, reflections and interferences through a layered material [SFCD99, IA00, HKYM00, HKYM01] at the expense of a higher computational cost. In order to reduce the requirements of memory and computation, Iehl et al. [IP00] have developed a hierarchical model. They use an adaptive hierarchical representation for the spectral distribution. The appropriate hierarchical level to perform a given reflection is chosen based on some perceptual metrics. Raso et al. [RF91] have presented a piecewise polynomial approximation. All these methods are usually too costly compared to the standard reflection model. Peercy [Pee91], with a generalized linear approach, has proven that in general, three basis functions are sufficient for image synthesis.

The main problem of spectral rendering is the availability of the input data. In general, and in our context, only a trichromatic color description is provided. Several approaches have been introduced to retrieve a spectral distribution from colors. The idea is to find a good spectral representation for a metamer (i.e. a family of spectra that are projected to the same color). The three lowest Fourier functions [Gla89b] and exponential functions [SFCD99] have been proposed since their variations are similar to the ones of color matching functions. With this approach, the retrieved spectra can have negative parts. In order to have only positive coefficients, Smits [Smi99] introduced an approach based on linear combinations of the spectral distribution representing 7 basic metamers (white, cyan, magenta, yellow, red, green, blue). This approach is the closest related to our goal. Unfortunately,

¹We will use RGB in this document but our approach can be applied to similar trichromatic color spaces like XYZ.

none of the previous methods guarantees a coherent behavior with the current rendering schemes under commonly used color illuminants.

Some recent work on shaders [Sta99, SFDC00] and on BRDFs [GH03] has shown that complex reflections can be obtained without a fine spectral sampling. Also, Peercy [Pee91] has shown that only three components are sufficient in the general case. Thus, we believe that a color reflection approach is still the most suitable solution for image synthesis, if we can globally control the behavior of color reflection. Our goal is to provide a general approach that keeps the color definition of all material properties unchanged, but that better preserves the color information under non-standard color illuminants.

A.3 Bilinear Reflection Model

The transformation from a spectral distribution $S(\lambda)$ to a color representation (metamer) $[S_c]_{c \in \{r,g,b\}}$ is controlled by the color matching functions $\{\bar{b}_c(\lambda), c \in \{r,g,b\}\}$:

$$\forall c \in \{r,g,b\}, S_c = \langle \bar{b}_c, S \rangle,$$

where $\langle f, g \rangle = \int f g$ denotes the usual dot product for function spaces.

This can be interpreted as an orthogonal projection into a color basis $\{b_c(\lambda), c \in \{r,g,b\}\}$. With this assumption, we can convert any color (C_r, C_g, C_b) into a spectral representative of the corresponding metamer:

$$C(\lambda) = \sum_c C_c b_c(\lambda).$$

Thus, given a color-version of incoming lighting $(L_r(\mathbf{s} \leftarrow \boldsymbol{\omega}_i), L_g(\mathbf{s} \leftarrow \boldsymbol{\omega}_i), L_b(\mathbf{s} \leftarrow \boldsymbol{\omega}_i))$ and a color-version of a BRDF $(\rho_r(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i), \rho_g(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i), \rho_b(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i))$, we can reconstruct their spectral representatives:

$$\begin{aligned} \bar{L}(\lambda, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) &= \sum_c L_c(\mathbf{s} \leftarrow \boldsymbol{\omega}_i) b_c(\lambda) \\ \bar{\rho}(\lambda, \mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) &= \sum_c \rho_c(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) b_c(\lambda) \end{aligned} \quad (\text{A.1})$$

using the notation illustrated in Figure 3.1.

Using these spectral representatives, the reflection on the surface is given by $\bar{L}(\lambda, \mathbf{s} \rightarrow \boldsymbol{\omega}_o) = \bar{L}(\lambda, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) \bar{\rho}(\lambda, \mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i)$, and its corresponding color is

$$\bar{\mathbf{C}}(\mathbf{s} \rightarrow \boldsymbol{\omega}_o) = \begin{bmatrix} \langle \bar{L}(\cdot, \mathbf{s} \rightarrow \boldsymbol{\omega}_o), \bar{b}_r \rangle \\ \langle \bar{L}(\cdot, \mathbf{s} \rightarrow \boldsymbol{\omega}_o), \bar{b}_g \rangle \\ \langle \bar{L}(\cdot, \mathbf{s} \rightarrow \boldsymbol{\omega}_o), \bar{b}_b \rangle \end{bmatrix}. \quad (\text{A.2})$$

By combining the Equations A.2 and A.1, we obtain a bilinear expression, similar to [Zmu92]: $\forall c \in \{r,g,b\}$

$$\bar{C}_c(\mathbf{s} \rightarrow \boldsymbol{\omega}_o) = \sum_d \sum_e L_d(\mathbf{s} \leftarrow \boldsymbol{\omega}_i) \rho_e(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) k_{d,e}^c,$$

In this equation, the 27 coefficients $k_{d,e}^c$ are defined by

$$k_{d,e}^c = \langle b_d b_e, \bar{b}_c \rangle. \quad (\text{A.3})$$

In fact, for symmetry reasons

$$k_{d,e}^c = k_{e,d}^c, \quad (\text{A.4})$$

the degrees of freedom for selecting these coefficients are reduced to 18.

This Equation A.3 defines a new reflection model that can be expressed by a matrix multiplication:

$$\bar{\mathbf{C}}(\mathbf{s} \rightarrow \boldsymbol{\omega}_o) = \mathbf{R}(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) \begin{bmatrix} L_r(\mathbf{s} \leftarrow \boldsymbol{\omega}_i) \\ L_g(\mathbf{s} \leftarrow \boldsymbol{\omega}_i) \\ L_b(\mathbf{s} \leftarrow \boldsymbol{\omega}_i) \end{bmatrix}, \quad (\text{A.5})$$

where the matrix $\mathbf{R}(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i)$ is defined as

$$\mathbf{R}(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) = \sum_e \rho_e(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) \mathbf{K}_e, \quad (\text{A.6})$$

$$\text{with } \mathbf{K}_e = \begin{bmatrix} k_{e,r}^r & k_{e,g}^r & k_{e,b}^r \\ k_{e,r}^g & k_{e,g}^g & k_{e,b}^g \\ k_{e,r}^b & k_{e,g}^b & k_{e,b}^b \end{bmatrix}.$$

A.4 New Reflection Operators

In order to reduce the required changes in current rendering systems, we introduce a new definition for the multiplication and division operators on trichromatic colors, and use them to define our new reflection approach. This has also been similarly introduced by Bergner et al. [BDM09].

A.4.1 New Color-Space Operators

The principal advantage of the model detailed in Equation A.5 is that the reflectance property and the light can still commute, thanks to the symmetry property of the coefficients $k_{d,e}^c$. Hence, we can redefine a multiplication on colors (denoted by $\otimes$) by

$$\bar{\mathbf{C}} = \mathbf{C}^1 \otimes \mathbf{C}^2 = [\sum \mathbf{C}_e^1 \mathbf{K}_e] \mathbf{C}^2 = \mathbf{C}^2 \otimes \mathbf{C}^1.$$

Moreover, a division operator on colors (denoted by $\oslash$) can similarly be introduced:

$$\mathbf{C}^2 = \bar{\mathbf{C}} \oslash \mathbf{C}^1 = [\sum \mathbf{C}_e^1 \mathbf{K}_e]^{-1} \bar{\mathbf{C}}, \quad (\text{A.7})$$

when the corresponding matrix is not singular, that is,

$$\det [\sum \mathbf{C}_e \mathbf{K}_e] = 0.$$

This is a third degree polynomial equation with C_r , C_g and C_b as unknowns. It defines a curved surface in color space. However, we note that when all three $\mathbf{K}_e$ matrices are non-singular, we can divide by a purely red, green or blue color, which is not the case when using the classical component-wise color multiplication.² We will further discuss this issue in the particular case of the ISO RGB color base in Section A.5.3.

²This classical approach corresponds to three singular matrices, where $k_{d,e}^c = 1$ if $c = d = e$ and $k_{d,e}^c = 0$ otherwise.

A.4.2 New Color Reflectance Model

Equation A.5 can now be rewritten as

$$\bar{\mathbf{C}}(\mathbf{s} \rightarrow \boldsymbol{\omega}_o) = \begin{bmatrix} \rho_r(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) \\ \rho_g(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) \\ \rho_b(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) \end{bmatrix} \otimes \begin{bmatrix} L_r(\mathbf{s} \leftarrow \boldsymbol{\omega}_i) \\ L_g(\mathbf{s} \leftarrow \boldsymbol{\omega}_i) \\ L_b(\mathbf{s} \leftarrow \boldsymbol{\omega}_i) \end{bmatrix}. \quad (\text{A.8})$$

Using this formulation, the reflection computation remains a three-step process:

1. Evaluation of the absorption $[\rho_c(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i)]_{c \in \{r,g,b\}}$
2. Evaluation of the incident light $[L_c^i(\boldsymbol{\omega}_i)]_{c \in \{r,g,b\}}$
3. Computation of the reflection using Equation A.8.

To implement this reflection model in an existing rendering system, only the color multiplication operator has to be rewritten.

Note that in complex shading, most of the time is spent in the first reflection step (i.e., computing the reflectance color [GH03]) or in the second step (like for radiosity algorithms [Ash94]). Our approach requires changing only the third step (i.e. the multiplication of incoming light with the reflectance color), and thus introduces only a small overhead.

A.4.3 Possible Simplifications

Our new reflection model can be simplified in most cases. For the diffuse case, the matrix $\mathbf{R}$ (see Equation A.6) is now direction-independent. Hence, it can be evaluated once and stored as the material description.

Moreover, it is often the case that the BRDF directional properties are independent of the color component, that is, $\forall c \in \{r,g,b\} \rho_c(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) = f(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) \rho_c$. In such cases (e.g. Phong's specular lobe [Pho75]), a direction-independent matrix $\mathbf{R}$ can also be pre-computed, based on the reflection coefficients $[\rho_c]_{c \in \{r,g,b\}}$. Hence, the Equation A.5 can be simplified to

$$\bar{\mathbf{C}}(\mathbf{s} \rightarrow \boldsymbol{\omega}_o) = \rho(\mathbf{s} \rightarrow \boldsymbol{\omega}_o, \mathbf{s} \leftarrow \boldsymbol{\omega}_i) \mathbf{R} \begin{bmatrix} L_r(\mathbf{s} \leftarrow \boldsymbol{\omega}_i) \\ L_g(\mathbf{s} \leftarrow \boldsymbol{\omega}_i) \\ L_b(\mathbf{s} \leftarrow \boldsymbol{\omega}_i) \end{bmatrix}.$$

The coefficients $k_{d,e}^c$ give us an easy way to convert the color representation of a BRDF, and to extend the reflection model to more general cases.

A.5 Coefficients Estimation: $k_{d,e}^c$

Estimating these coefficients from a set of color matching functions is a two-step process. We first introduce a coherent behavior condition that allows our approach to match the required reflection behavior for a *constant* light source. We then present how to compute an approximation based on a regular sampling of the color matching functions. This first approximation is then minimized in order to fit the coherent behavior condition.

A.5.1 Coherent Behavior Condition

As previously stated, when the color matching functions are normalized, the regular reflection model provides the correct reflected color under a *constant* illuminant (i.e., a white or gray

illuminant)[Bor91]. For a user point of view, this means that the reflected color is equal to the surface reflectance under a *constant* light. Even if the color space is not normalized, we want to maintain this property in order to provide a behavior that is coherent with the current model. The reflectance multiplied by a *constant* light is simply itself, but scaled by the light intensity, leading to the following equation (with $\mathbf{I}$ being the identity matrix):

$$\mathbf{K}_r + \mathbf{K}_g + \mathbf{K}_b = \mathbf{I}. \quad (\text{A.9})$$

A.5.2 Estimation from Color Matching Functions

For this evaluation, we approximate the color matching functions by:

$$\bar{b}_c(\lambda) \simeq \sum_{j=1}^n \bar{c}_j \phi_j(\lambda) \quad (\text{A.10})$$

where $(\bar{c}_i, i \in [1..n], c \in \{r, g, b\})$ are the regular sampled values of the matching functions - like those provided by the CIE[CIE], and $(\phi_i(\lambda), i \in [1..n])$ is an interpolation basis.

If we want the projection to the color basis to be orthogonal, the color basis has to be a linear combination of the color matching functions:

$$\begin{bmatrix} b_r \\ b_g \\ b_b \end{bmatrix} = \mathbf{M} \begin{bmatrix} \bar{b}_r \\ \bar{b}_g \\ \bar{b}_b \end{bmatrix}. \quad (\text{A.11})$$

Note that, since the projection is orthogonal, the spectral representatives defined in Equation A.1 minimize the $\mathcal{L}^2$ distance to all the possible spectral distributions of the metamer.

As these two bases are duals, that is $\langle \bar{b}_i, b_j \rangle = \delta_{i,j}$, we have:

$$\mathbf{M}^{-1} = \begin{bmatrix} \langle \bar{b}_r, \bar{b}_r \rangle & \langle \bar{b}_r, \bar{b}_g \rangle & \langle \bar{b}_r, \bar{b}_b \rangle \\ \langle \bar{b}_g, \bar{b}_r \rangle & \langle \bar{b}_g, \bar{b}_g \rangle & \langle \bar{b}_g, \bar{b}_b \rangle \\ \langle \bar{b}_b, \bar{b}_r \rangle & \langle \bar{b}_b, \bar{b}_g \rangle & \langle \bar{b}_b, \bar{b}_b \rangle \end{bmatrix}. \quad (\text{A.12})$$

We have now an approximation of the color basis and its dual, and hence we can compute the coefficients $\tilde{k}_{d,e}^c$ using Equation A.3. They will be a first approximation of the final coefficients $k_{d,e}^c$. Note that, by construction, these coefficients follow the symmetry condition (see Equation A.4).

However, due to numerical instabilities, the coefficients may not follow the coherent behavior condition (see Equation A.9), even for normalized color matching functions. Therefore, we minimize them under the coherent behavior condition to compute the closest satisfying coefficients $k_{d,e}^c$.

A.5.3 Estimation for ISO RGB Color Space

For our purpose, we use the ISO RGB color space that is becoming a standard (see Figure A.1 and [ISO21]). Since it is an *Unrendered Color Space*, there are no specified dynamic ranges or viewing conditions, making it suitable for lighting computations. Furthermore, the color matching functions are normalized, implicitly ensuring the coherent behavior condition.

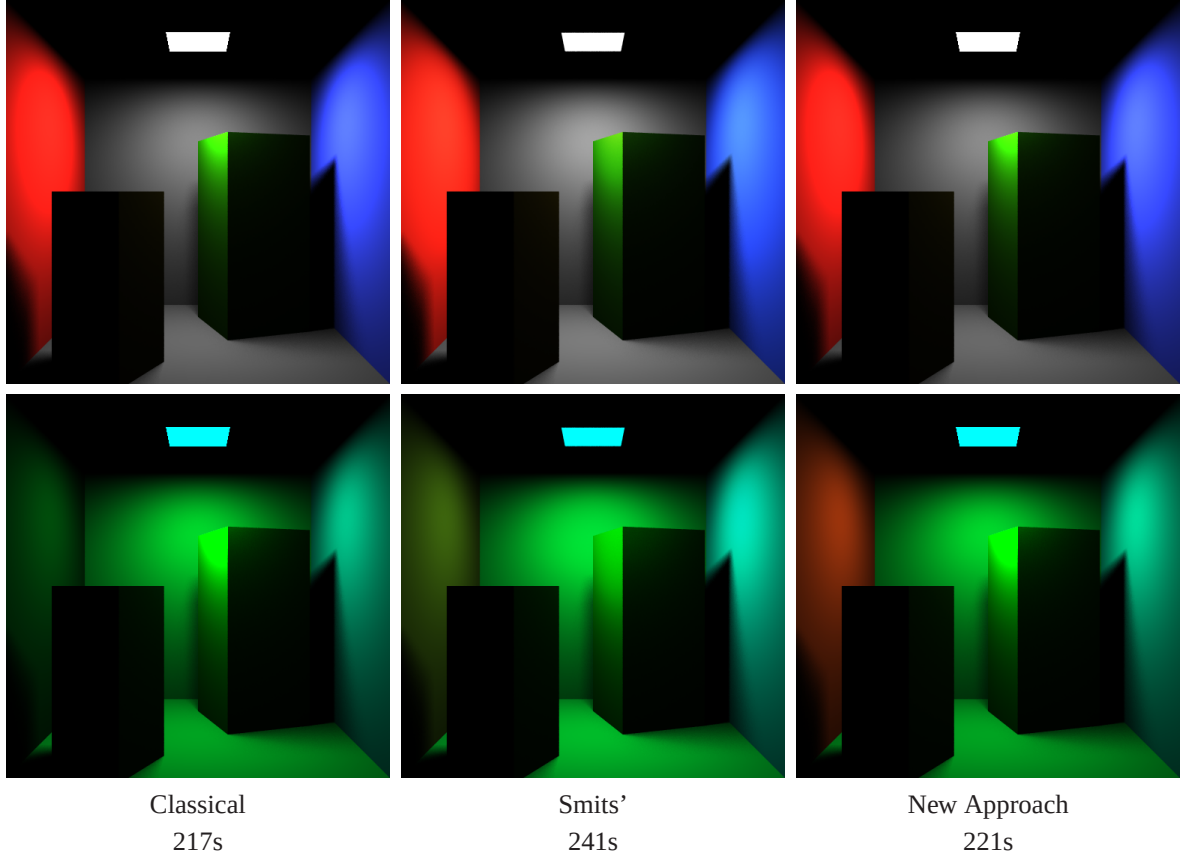


Fig. A.3: Direct lighting of a Cornell box. The first row corresponds to the solutions under a *constant* illuminant, the second one under a *cyan* illuminant.

In the following, we present the coefficients obtained for the ISO RGB color space in a matrix notation:

$$\begin{aligned}
 \mathbf{K}_r &= \begin{bmatrix} 0.6533043776, & 0.2708567778, & 0.0758388444 \\ 0.0594368555, & -0.0413179278, & -0.0181189277 \\ -0.0111905666, & 0.0005536667, & 0.0106368999 \end{bmatrix} \\
 \mathbf{K}_g &= \begin{bmatrix} 0.2708567778, & -0.2135258222, & -0.0573309555 \\ -0.0413179278, & 0.9943860889, & 0.0469318388 \\ 0.0005536667, & -0.0401482999, & 0.0395946333 \end{bmatrix} \\
 \mathbf{K}_b &= \begin{bmatrix} 0.0758388444, & -0.0573309555, & -0.0185078889 \\ -0.0181189277, & 0.0469318388, & -0.0288129111 \\ 0.0106368999, & 0.0395946333, & 0.9497684667 \end{bmatrix}.
 \end{aligned}$$

Note that the determinants of the matrices $\mathbf{K}_r$, $\mathbf{K}_g$ and $\mathbf{K}_b$ are non-zero. This does not ensure that the division operator (see Equation A.7) is always possible, but as pointed out in Section A.4.1, this case is restricted to a surface in color space. For quantized colors, such as the standard 8-bit color definition, it only occurs at $(0, 0, 0)$ color.

A.6 Reflection Behavior Control

A.6.1 Coefficients Meaning

In the coefficients $k_{d,e}^c$ for the ISO RGB color matching functions, we can notice that the largest ones occur at $c = d = e$. For the classical reflection behavior, their value is 1, and the other coefficients are 0, which corresponds to the case where no transfer occurs from one color component to another. Using our reflection operators, in the ISO RGB case, most of the transfer occurs to the red color component (see Section A.7). Note that $k_{r,r}^r$ is also the smallest $k_{d,e}^c$ coefficient.

So intuitively, the $k_{d,e}^c$ with $c = d = e$ corresponds to the amount of energy that is not transferred from one color component c and reciprocally, $1 - k_{c,c}^c$ corresponds to the amount of energy that is transferred to this color component.

Note also that the coefficients, which control the transfer to the component c , are $k_{c,c}^c$. Only those have to be modified in order to change the reflection behavior of the component c .

A.6.2 Basis-Dependent Coefficients Adjustment

Hence, one way to control the behavior of the reflection is to select the amount of energy that is transferred to a component $\alpha_c, c \in \{r, g, b\}$. Based on the previous observations, we can simply adjust the coefficients $k_{c,c}^c$ as follows:

$$k'_{d,e}{}^c = \begin{cases} 1 - \alpha_c, & \text{if } c = d = e \\ \beta_c k_{d,e}^c, & \text{otherwise} \end{cases}.$$

This modification preserves the symmetry condition (see Equation A.4) since

$$k'_{d,e}{}^c = \beta_c k_{d,e}^c = \beta_c k_{d,e}^c = k'_{d,e}{}^c.$$

The new coefficients also have to follow the coherent behavior condition, as defined by Equation A.9. This leads to choose the following values of β_c :

$$\beta_c = \frac{\alpha_c}{1 - k_{c,c}^c}, \forall c \in \{r, g, b\}.$$

Note that, when $\alpha_c = 0$ (i.e., there is no transfer to the c th component), then $\beta_c = 0$. Hence, adjusting the α_c allows a smooth transition from the standard reflection model to our approach.

A.6.3 User-Defined Coefficients

The basis dependent control introduces a different behavior for each color component, since it is only an adjustment of pre-defined coefficients. From a user point of view, it is more intuitive if there is a similar behavior for each component. For this purpose, we introduce a new set of coefficients that are not dependent on a particular color definition of coefficients. They are based on the same idea of controlling the amount of energy that is transferred to a component $\alpha_c, c \in \{r, g, b\}$. We define

$$\mathbf{K}_r = \begin{bmatrix} 1 - \alpha_r & \beta_r & \beta_r \\ \gamma_g & \beta_g & \gamma_g \\ \gamma_b & \gamma_b & \beta_b \end{bmatrix}, \mathbf{K}_g = \begin{bmatrix} \beta_r & \gamma_r & \gamma_r \\ \beta_g & 1 - \alpha_g & \beta_g \\ \gamma_b & \gamma_b & \beta_b \end{bmatrix},$$

$$\mathbf{K}_b = \begin{bmatrix} \beta_r & \gamma_r & \gamma_r \\ \gamma_g & \beta_g & \gamma_g \\ \beta_b & \beta_b & 1 - \alpha_b \end{bmatrix}, \text{ with } \begin{cases} \beta_i = \frac{\alpha_i}{2} \\ \gamma_i = -\frac{\alpha_i}{4} \end{cases}.$$

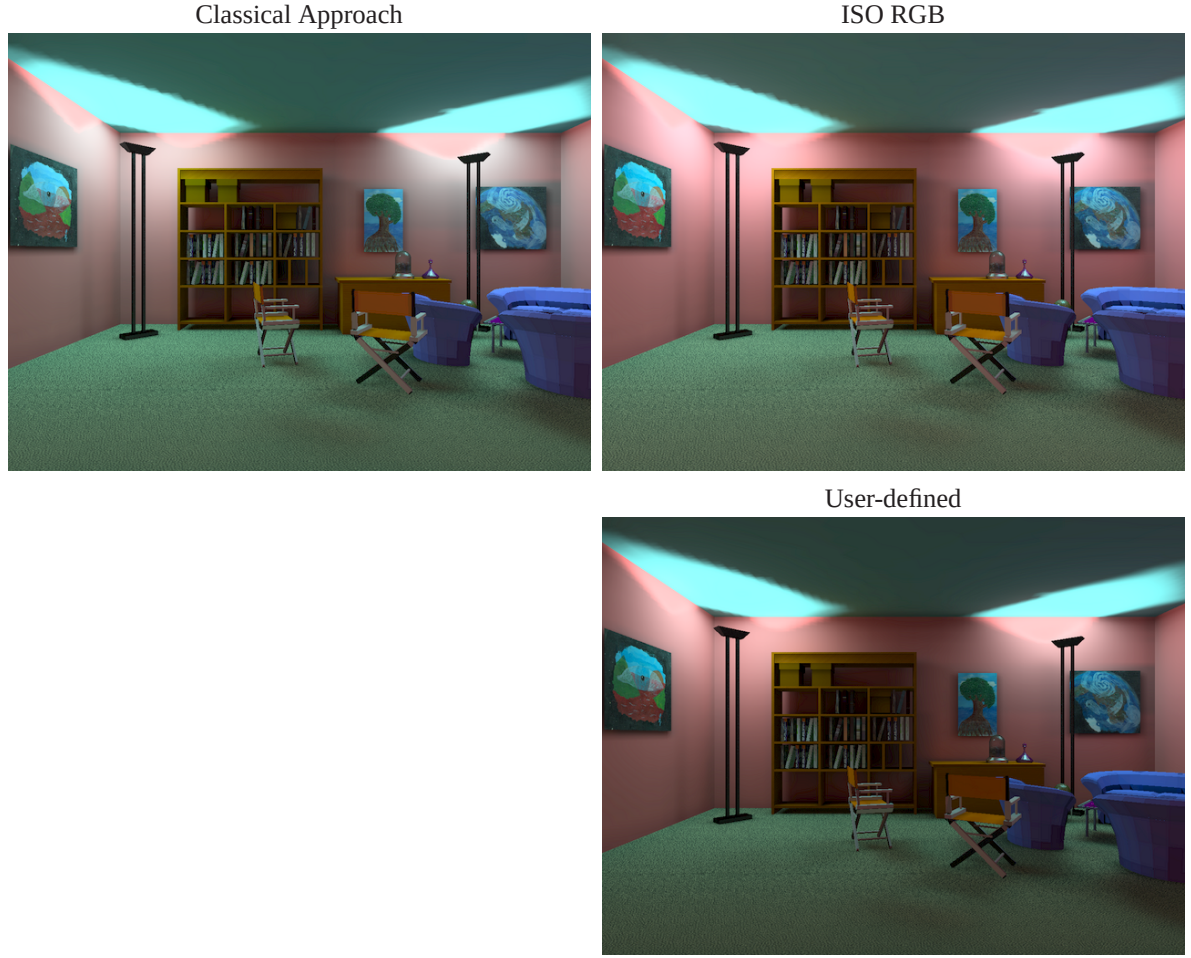


Fig. A.4: The "psychedelic" lounge illuminated. Note that the overall redness is better preserved in our approach. This is even more noticeable leftmost painting.

These coefficients follow the symmetry condition (see Equation A.4) and the coherent behavior condition (see Equation A.9).

A.7 Applications and Results

A.7.1 Global Illumination

We have tested our new approach for two software illumination solutions. For the first one, we computed only the direct lighting, and for the second one, the full diffuse global lighting, for which the multiple bounces can create strong indirect lighting effects that are difficult to control.

For the direct lighting scenario, we used a simple ray-tracer. The $k_{d,e}^c$ coefficients from ISO RGB color-matching functions are simply used to convert the color definition of surfaces into a matrix representation, as described in the Section A.4.3. For a comparison, we also implemented Smits' [Smi99] approach to convert the initial RGB data to a full spectral definition of materials, and the results are shown in Figure A.3. Note that our new technique has only a small impact (2% more) on



Fig. A.5: Adjusting the amount of transfer to the blue component. The coefficients are the same as in Figure A.4, with the transfer to blue set to 100%.

the rendering time³. With more complex shaders, the relative impact on the performance is smaller, since our approach does not change the way reflectance is computed, but only the final multiplication between the incoming light and the estimated reflectance.

By construction, under a *constant* illuminant, our approach provides the same solution as the classical model, whereas the spectral solution creates a larger highlight on the red wall. This is due to the fact that the coherent behavior condition (see Section A.5.1) is not respected. Under a *cyan* illuminant, our approach preserves the red aspect of the wall that completely disappears with the classical reflection.

Most of the loss in color may appear in indirect lighting. To demonstrate this effect, we designed a "psychedelic" lounge 3D scene for which we computed a radiosity solution. Note that for the implementation of our approach, we had to rewrite only the multiplication operator for colors. In this scene, the light sources are facing the *blue* ceiling. The indirect light is therefore mostly blue. As shown in Figure A.4, our approach preserves the color appearance whereas the classical approach creates a more grayish aspect. This is more noticeable on the walls and on the leftmost painting on the walls, for which the red color is more saturated. We also used the basis-independent design to compute the solution with the same transfer amount than for the ISO RGB ones (i.e., $\alpha_r = 35\%$, $\alpha_g = 1\%$ and $\alpha_b = 5\%$, see Sections A.6.3 and A.5.3). Even if the results are quite similar, the red component is more saturated for the ISO RGB case, since the red color matching function is quite different from the green and blue ones. In the Basis-independent case, the behavior is similar for the three components by design.

By adjusting the transfer amount to blue to 100%, we can control the general bluish aspect of the scene (see Figure A.5). We can notice that the resulting solution has a more bluish aspect than in the Figure A.4, without modifying any material property. As the lighting is mostly indirect, it will be difficult to obtain the same behavior with another approach.

A.7.2 Simple Relighting

We also used our reflection model to implement a basic relighting approach, with a simple change of the color of the light source. Working on high-dynamic range images [DM97], the relighting process

³on an Intel Pentium4 2.6 GHz with 1 GB of memory

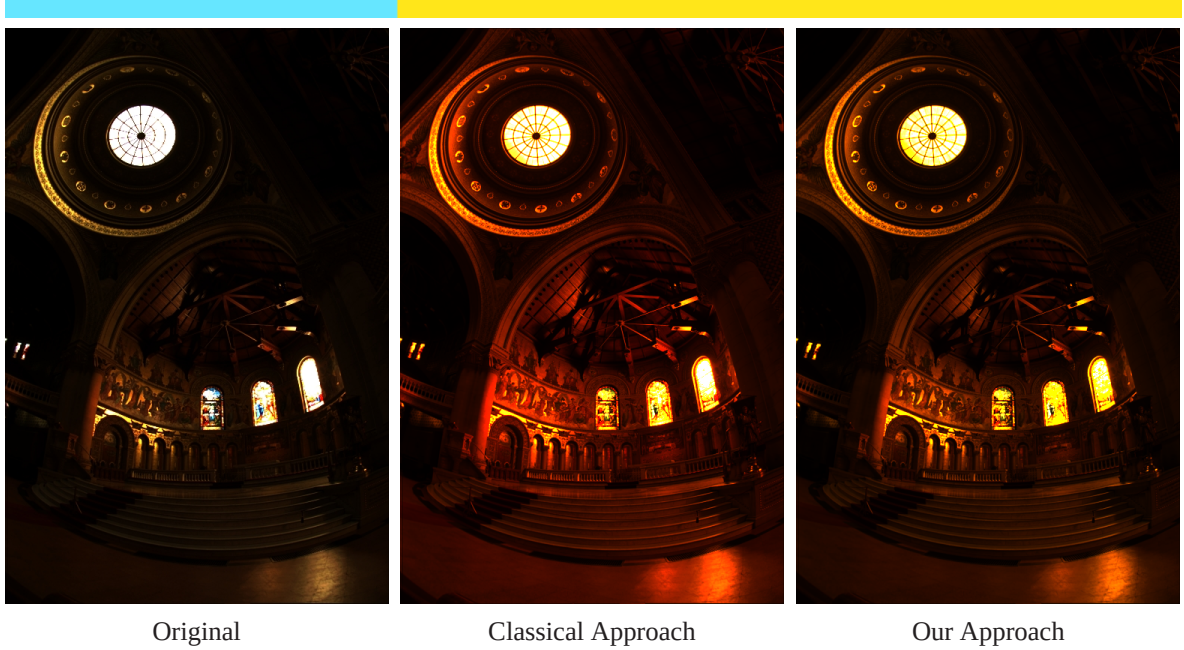


Fig. A.6: Changing outdoor light on the memorial. Note the over-saturated red in the classical approach.

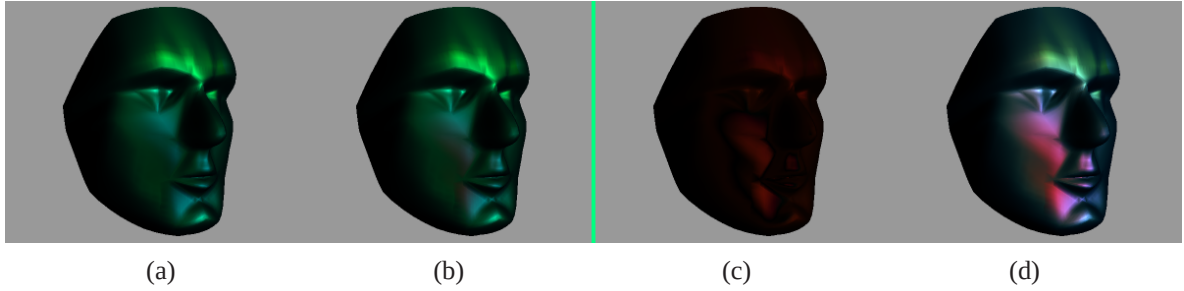


Fig. A.7: Hardware implementation: (a) regular reflection – (b) our approach – (c) difference – (d) reflection under a *white* light source. For (a) and (b), the color of the light source is shown in the middle rectangle. Note that the red highlight on the cheek (d) has disappeared using the classical approach (a) and it is still present using our approach.

is simply the division of the pixel color by the previous illuminant and its multiplication with the new illuminant. We have experimented a change from a blue-sky light source $(0.4, 0.9, 1.0)$ to a yellow illuminant $(1.0, 0.9, 0.1)$, using the ISO RGB coefficients. As can be seen in Figure A.6, the color information is better preserved using our approach.

A.7.3 Hardware Accelerated Rendering

As the last application, we have implemented our approach with ISO RGB coefficients using programmable graphics hardware⁴. For this purpose, we use the iridescent BRDF model [GH03] in order to show a component-dependent behavior. The classical solution to increase color preservation is to represent a material property by a 3×3 matrix. Hence, for a component-dependent BRDF, 9 functions have to be defined. Using our approach, we keep the 3-component color definition, and simply add

⁴ARB_{vertex/fragment}_program OpenGL extensions on an NVIDIA Quadro FX 500

11 instructions at the end of the fragment program for the reflection evaluation. The original shader is composed of a fragment program with 42 instructions and a vertex program with 70 instructions. This has only a small impact on the frame-rate (160 fps instead of 170 fps for a 400x400 image).

The results are shown in Figure A.7. Once again, the color appearance is better preserved. The red reflection is still present in our approach whereas it disappeared in the classical solution.

A.8 Conclusion

We have introduced new multiplication and division operators adapted to the trichromatic color representation, in order to better preserve the appearance color in reflections. This new approach requires a minimal change to be implemented in current rendering systems, and behaves coherently with respect to existing solutions. This coherent behavior allows the users to keep the same approach when designing the appearance of surfaces.

We have also presented two approaches for controlling the reflection behavior resulting from these operators. This is done by an adaptation of the original coefficients based on color matching functions, or by an independent design, with the same behavior for the RGB color components. This control is done by adjusting the amount of energy that is transferred to a selected component.

As examples, we have implemented our solution in three possible applications: global illumination, simple relighting and hardware rendering. We have also experimented different set of coefficients, both from the ISO RGB color matching functions, and from our basis-independent design. The results have shown a better behavior in preserving color information with a low overhead in implementation and computation.

Future Work

This work can be extended in two directions, namely the computation of the matrix coefficients and the user interface. Concerning the coefficient computation, we have to find a new basis-independent design that would allow to control the transfer from one color component to another, instead of the overall transfer from all components to one. Such an approach allows a more precise and complete control of the reflection behavior.

Concerning the user interface, some studies have to be done on how the user expects the scene to appear under different lighting conditions using perceptual studies. This demands a comparison with the results obtained through classical color constancy approaches [FDF94]. Furthermore, we have to find a solution to reduce the apparition of negative values during the reflection.

Vectorial Definitions

In this annex, we present the current state of our investigations on using vector representations for computing global illumination. We also present some ideas about improved polynomial bases for such directional distributions.

B.1 Definitions

Before going into details about the representation, we need to introduce carefully the different terms and notation that we use in this annex.

B.1.1 Hemispherical Function

For most of the reflection phenomena, only lights coming from the upper hemisphere (i.e., the portion of visible direction on a surface) are taken into account. An accurate representation of such a hemisphere is somewhat difficult with a large set of direction bases [Kř04], and this leads to the development of new Hemispherical Spherical Harmonics [GKPB04].

In the entire document, we have defined the hemispherical function h around direction $\mathbf{n}$ by

$$\Omega \times \Omega \rightarrow \mathbb{R} : h(\mathbf{n}, \boldsymbol{\omega}) = \begin{cases} 1 & \text{if } {}^T\mathbf{n}\boldsymbol{\omega} \geq 0 \\ 0 & \text{otherwise} \end{cases} \quad (\text{B.1})$$

where ${}^T\mathbf{n}$ is the transposed normal.

B.1.2 Directional Radiance Vector

This new function contains in one unique value both the direction and the energy of the lighting:

$$\Omega \rightarrow \mathbb{R}^3 : \mathbf{L}(\boldsymbol{\omega}) = L(\boldsymbol{\omega}) \boldsymbol{\omega}.$$

One may think of that directional information is redundant this contained in both the parameter $\boldsymbol{\omega}$ and in the normalized value since $\mathbf{L}(\boldsymbol{\omega})/|\mathbf{L}(\boldsymbol{\omega})| = \boldsymbol{\omega}$. In the following, we will see that, when projected in some given bases, this equality is actually now false.

B.1.3 Directional Irradiance Vector

We have already introduced this representation in Chapter 3 - Section 3.2.1. We re-defined *directional irradiance vector* $\mathbf{I}(\mathbf{n})$ here using the directional radiance vector (with $\boldsymbol{\omega}$ as a direction and Ω as the space of all directions that is, a unit sphere):

$$\Omega \rightarrow \mathbb{R}^3 : \mathbf{I}(\mathbf{n}) = \int_{\Omega} \mathbf{L}(\boldsymbol{\omega}) h(\mathbf{n}, \boldsymbol{\omega}) d\boldsymbol{\omega}. \quad (\text{B.2})$$

This is exactly similar to the *irradiance vector* [Arv94], with as a main difference the fact that we preserve the fact that this vector is normal-dependent. The classical irradiance I for a normal $\mathbf{n}$ is thus defined as a classical dot product

$$\begin{aligned} I &= \mathbf{n}^T \mathbf{I}(\mathbf{n}) \\ &= \mathbf{n}^T \int_{\Omega} \mathbf{L}(\boldsymbol{\omega}) h(\mathbf{n}, \boldsymbol{\omega}) d\boldsymbol{\omega} \\ &= \mathbf{n}^T \int_{\Omega} L(\boldsymbol{\omega}) \boldsymbol{\omega} h(\mathbf{n}, \boldsymbol{\omega}) d\boldsymbol{\omega} \\ &= \int_{\Omega} L(\boldsymbol{\omega}) (\mathbf{n}^T \boldsymbol{\omega}) h(\mathbf{n}, \boldsymbol{\omega}) d\boldsymbol{\omega}. \end{aligned}$$

B.1.4 Better Compression of Directional Vectors

Remember that Equation B.2 defines an irradiance vector (requiring three floating point values) for a single wavelength. Since, in computer graphics a color is defined by three primary colors (R, G, B), we need three irradiance vectors stored in a 3×3 matrix $\mathbf{M} = [\mathbf{I}_R \ \mathbf{I}_G \ \mathbf{I}_B]$. We compress $\mathbf{M}$ as the product of a direction $\mathbf{d}$ and a color $\mathbf{c}$ ($\mathbf{M} = \mathbf{d} \mathbf{c}^T$) defined as follows:

$$\mathbf{d} = \frac{\mathbf{I}_R + \mathbf{I}_G + \mathbf{I}_B}{|\mathbf{I}_R + \mathbf{I}_G + \mathbf{I}_B|} \quad \mathbf{c} = \begin{bmatrix} |\mathbf{I}_R| \\ |\mathbf{I}_G| \\ |\mathbf{I}_B| \end{bmatrix}.$$

This first approximation does not guarantee that we obtain the best choice of $\mathbf{d}$ and $\mathbf{c}$. Given the fact that $\mathbf{d}$ is a pure direction, it can be proven that the best mean square approximation has to fulfill the following equations

$$\begin{cases} |\mathbf{d}| = 1 \\ |\mathbf{c}|^2 \mathbf{d} - \mathbf{M} \mathbf{c} = 0 \\ |\mathbf{d}|^2 \mathbf{c} - \mathbf{M}^T \mathbf{d} = 0 \end{cases}$$

This leads to the following iterative process that has experimentally shown some improvements in the approximation error:

$$\begin{cases} \mathbf{d}_0 = \mathbf{d} \\ \mathbf{c}_0 = \mathbf{c} \end{cases} \quad \begin{cases} \mathbf{d}_{n+1} = \frac{\mathbf{M} \mathbf{c}_n}{|\mathbf{M} \mathbf{c}_n|} \\ \mathbf{c}_{n+1} = \mathbf{M}^T \mathbf{d}_n \end{cases}$$

B.2 Bases for Directional Vectors

B.2.1 Scalar directional basis

In this section, we consider using classical directional basis function, such as spherical harmonics or wavelets. While using such an approach relies on well-known bases, it requires storing complex vector coefficients (cf. Section B.1.4). We study the differences in-between the projection of directional irradiance vector with the projection of directional radiance vector.

Scalar Basis for Directional Irradiance Vector

For a given directional basis $\{\Omega \rightarrow \mathbb{R} : \phi_i(\boldsymbol{\omega})\}$, we approximate directional irradiance vector using its dual basis $\{\Omega \rightarrow \mathbb{R} : \phi_i^*(\boldsymbol{\omega})\}$:

$$\begin{aligned} \mathbf{I}(\mathbf{n}) &\simeq \bar{\mathbf{I}}(\mathbf{n}) = \sum_i \mathbf{I}_i \phi_i(\mathbf{n}) \\ &= \sum_i \langle \mathbf{I}(\mathbf{n}), \phi_i^*(\mathbf{n}) \rangle \phi_i(\mathbf{n}), \end{aligned}$$

where $\langle f, g \rangle = \int f g$ denotes the usual dot product for space of functions $\Omega \rightarrow \mathbb{R}$.

Interestingly, while developing the expression of $\mathbf{I}_i$, we obtain the following:

$$\begin{aligned} \mathbf{I}_i &= \langle \mathbf{I}(\mathbf{n}), \phi_i^*(\mathbf{n}) \rangle \\ &= \int_{\Omega} \mathbf{I}(\mathbf{n}) \phi_i^*(\mathbf{n}) d\mathbf{n} \\ &= \int_{\Omega} \int_{\Omega} \mathbf{L}(\boldsymbol{\omega}) h(\mathbf{n}, \boldsymbol{\omega}) \phi_i^*(\mathbf{n}) d\mathbf{n} d\boldsymbol{\omega}. \end{aligned}$$

Introducing the new functions

$$\Omega \rightarrow \mathbb{R} : \psi_i^*(\boldsymbol{\omega}) = \int_{\Omega} h(\mathbf{n}, \boldsymbol{\omega}) \phi_i^*(\mathbf{n}) d\mathbf{n}.$$

this reduces to

$$\begin{aligned} \mathbf{I}_i &= \int_{\Omega} \mathbf{L}(\boldsymbol{\omega}) \psi_i^*(\boldsymbol{\omega}) d\boldsymbol{\omega} \\ &= \langle \mathbf{L}(\boldsymbol{\omega}), \psi_i^*(\boldsymbol{\omega}) \rangle. \end{aligned}$$

These new functions ψ_i^* , projections of the hemisphere in the chosen basis, thus directly project directional radiance vectors into approximated directional irradiance vectors.

Scalar Basis for Directional Radiance Vector

Using the same directional basis $\{\Omega \rightarrow \mathbb{R} : \phi_i(\boldsymbol{\omega})\}$ and its dual $\{\Omega \rightarrow \mathbb{R} : \phi_i^*(\boldsymbol{\omega})\}$, we can project the directional radiance vector as following:

$$\begin{aligned} \mathbf{L}(\boldsymbol{\omega}) &\simeq \bar{\mathbf{L}}(\boldsymbol{\omega}) = \sum_i \mathbf{L}_i \phi_i(\boldsymbol{\omega}) \\ &= \sum_i \langle \mathbf{L}(\boldsymbol{\omega}), \phi_i^*(\boldsymbol{\omega}) \rangle \phi_i(\boldsymbol{\omega}). \end{aligned}$$

With such an approximation, since each coefficient L_i corresponds to a weighted average of directional vector radiance on the support of basis function ϕ_i , there is no more any guaranty that $\mathbf{L}(\boldsymbol{\omega})/|\mathbf{L}(\boldsymbol{\omega})| = \boldsymbol{\omega}$. This is the main advantage of such representation since it precisely adjusts each direction to a new one that better corresponds to the main incoming lighting direction.

Using this approximation for directional radiance vector, the resulting directional vector irradiance is computed as

$$\begin{aligned}\tilde{\mathbf{I}}(\mathbf{n}) &= \int_{\Omega} \bar{\mathbf{L}}(\boldsymbol{\omega}) h(\mathbf{n}, \boldsymbol{\omega}) \boldsymbol{\omega} \\ &= \sum_i L_i \int_{\Omega} \phi_i(\boldsymbol{\omega}) h(\mathbf{n}, \boldsymbol{\omega}) d\boldsymbol{\omega}.\end{aligned}$$

We introduce a new function, similarly to the previous section:

$$\Omega \rightarrow \mathbb{R} : \psi_i(\mathbf{n}) = \int_{\Omega} h(\mathbf{n}, \boldsymbol{\omega}) \phi_i(\boldsymbol{\omega}) d\boldsymbol{\omega}.$$

ψ_i is the projection of the hemisphere function in the dual of the given basis. Note that ψ and ψ^* are not dual and that if ϕ is an orthonormal basis, $\psi = \psi^*$. With this new function, the directional vector irradiance is also directly approximated by

$$\tilde{\mathbf{I}}(\mathbf{n}) = \sum_i L_i \psi_i(\mathbf{n})$$

One unique set of coefficients represents the directional vector radiance and irradiance.

B.2.2 Vector Basis

In previous section, the vector information is supported by the coefficients. Another approach is to remove the directional information from the coefficient by directly using a vector basis such as Vector Spherical Harmonics [Hil54]. We explore such a solution in this section.

Vector Basis for Directional Irradiance Vector

For a given directional vector basis $\{\Omega \rightarrow \mathbb{R}^3 : \phi_i(\boldsymbol{\omega})\}$, we approximate directional irradiance vector using its dual $\{\Omega \rightarrow \mathbb{R}^3 : \phi_i^*(\boldsymbol{\omega})\}$:

$$\begin{aligned}\mathbf{I}(\mathbf{n}) &\simeq \bar{\mathbf{I}}(\mathbf{n}) = \sum_i I_i \phi_i(\mathbf{n}) \\ &= \sum_i \langle \mathbf{I}(\mathbf{n}), \phi_i^*(\mathbf{n}) \rangle \phi_i(\mathbf{n}),\end{aligned}$$

where $\langle \mathbf{f}, \mathbf{g} \rangle = \int \mathbf{f}^T \mathbf{g}$ is the extension of usual dot product for space of directional scalar distributions $\Omega \rightarrow \mathbb{R}$ to space of directional vector distributions $\Omega \rightarrow \mathbb{R}^3$. As expected, I_i is now a scalar value.

Similarly to Section B.2.1, we define a corresponding projection function ψ_i^* from directional radiance vector to directional incident radiance vector:

$$\Omega \rightarrow \mathbb{R}^3 : \psi_i^*(\boldsymbol{\omega}) = \int_{\Omega} h(\mathbf{n}, \boldsymbol{\omega}) \phi_i^*(\mathbf{n}) d\mathbf{n}$$

ψ_i^* is also the projection of the hemisphere function in the given vector basis. The approximation coefficients of the directional irradiance vector are thus:

$$I_i = \int_{\Omega} \mathbf{L}(\boldsymbol{\omega}) \psi_i^*(\boldsymbol{\omega}) d\boldsymbol{\omega}$$

Vector Basis for Directional Radiance Vector

Using the same directional basis $\{\Omega \rightarrow \mathbb{R}^3 : \phi_i(\omega)\}$ and its dual $\{\Omega \rightarrow \mathbb{R}^3 : \phi_i^*(\omega)\}$, we can project the directional radiance vector as following:

$$\begin{aligned} \mathbf{L}(\omega) &\simeq \bar{\mathbf{L}}(\omega) = \sum_i L_i \phi_i(\omega) \\ &= \sum_i \langle \mathbf{L}(\omega), \phi_i^*(\omega) \rangle \phi_i(\omega). \end{aligned}$$

One again, L_i is now a classical scalar value.

Using the same approach than in Section B.2.1, we introduce the function

$$\Omega \rightarrow \mathbb{R} : \psi_i(n) = \int_{\Omega} h(n, \omega) \phi_i(\omega) d\omega,$$

resulting in a direct approximation of the directional irradiance vector using the same coefficients that for the directional radiance vector:

$$\tilde{\mathbf{I}}(n) = \sum_i L_i \psi_i(n)$$

ψ_i is now the projection of the hemisphere function in the dual of the given vector basis. Note that ψ and ψ^* are not dual and that if ϕ is an orthonormal basis, $\psi = \psi^*$.

One easy way to define a vector basis is to set $\phi_i^*(\omega) = \phi_i(\omega)\omega$. Using such basis functions leads to the followings:

$$\begin{aligned} L_i &= \int_{\Omega} {}^T\mathbf{L}(\omega) \phi_i^*(\omega) d\omega \\ &= \int_{\Omega} L(\omega) \phi^*(\omega) {}^T\omega d\omega \\ &= \int_{\Omega} L(\omega) \phi^*(\omega) d\omega. \end{aligned}$$

This corresponds to the radiance coefficient of a classical directional distribution. Thus, such vector basis does not introduce any advantages for representing directional vector radiance.

B.2.3 Comparisons

Based on this analysis, both approaches seem to be similar for direction irradiance vector. But it seems that it is better to use a scalar basis for representing directional radiance vector. Furthermore, when projecting directional radiance vector into a given basis, the same coefficients can be used for reconstructing both the radiance and the irradiance. This leads to a preferred choice to scalar direction bases.

This approach would also possibly improve the following. With a classical directional representation, we need a fine subdivision in order to capture the directional variations. With vector approach, the “mean” direction is directly embedded in the coefficients, and this provides useful information about the directionality of the phenomena. If a directional distribution is still required, fewer coefficients would be required.

B.3 Extending Radiosity Equation

Using vector intermediate representation has been proved as improving the quality of radiosity algorithms [WHG99, LLAP05]. In this section, we explore the full extension of radiosity equation to vector representation.

B.3.1 Radiosity Vector

In a diffuse environment, the radiosity is defined as the reflected irradiance $B = rI$. Similarly to the directional irradiance vector, we thus define the *directional radiosity vector* as

$$\mathbf{B}(\mathbf{n}) = r\mathbf{I}(\mathbf{n}) = r \int_{\Omega} L(\boldsymbol{\omega}) \boldsymbol{\omega} h(\mathbf{n}, \boldsymbol{\omega}) d\boldsymbol{\omega}. \quad (\text{B.3})$$

Similar to irradiance vector, we also have

$$B = {}^T\mathbf{n}\mathbf{B}(\mathbf{n}). \quad (\text{B.4})$$

For the following, since we move in a discretized environment, composed of planar patches (i.e., the normal is constant on a patch), we only use $\mathbf{B}_i$ for $\mathbf{B}_i(\mathbf{n})$ and $\mathbf{I}_i$ for $\mathbf{I}_i(\mathbf{n})$

B.3.2 Interaction Matrix

For a given discretization of the 3D scene, the radiosity equation is expressed as

$$B_i = E_i + r_i \sum_j F_{ij} B_j, \quad (\text{B.5})$$

where E_i is the diffuse emissivity of the patch S_i and F_{ij} is the *form-factor* which controls the proportion of energy emitted by a patch S_j and received on a patch i . This form-factor is defined as

$$F_{ij} = \frac{1}{A_i} \int_{S_i} \int_{S_j} V(\mathbf{s}_i, \mathbf{s}_j) \frac{{}^T\mathbf{n}_i \mathbf{s}_{ij} {}^T\mathbf{n}_j \mathbf{s}_{ji}}{\pi |\mathbf{s}_{ij}|^4} d\mathbf{s}_i d\mathbf{s}_j, \quad (\text{B.6})$$

where $\mathbf{s}_{ij} = \mathbf{s}_j - \mathbf{s}_i$. In this equation, V is the visibility function. It is defined as

$$V(\mathbf{s}_i, \mathbf{s}_j) = \begin{cases} 0 & {}^T\mathbf{n}_i \mathbf{s}_{ij} \leq 0 \\ 0 & {}^T\mathbf{n}_j \mathbf{s}_{ji} \leq 0 \\ 0 & \mathbf{s}_i \text{ is not visible from } \mathbf{s}_j \\ 1 & \text{otherwise} \end{cases} \quad (\text{B.7})$$

We thus rewrite the Equation B.6 as

$$\begin{aligned} F_{ij} &= {}^T\mathbf{n}_i \left(\frac{1}{A_i} \int_{S_i} \int_{S_j} V(\mathbf{s}_i, \mathbf{s}_j) \frac{\mathbf{s}_{ij} {}^T\mathbf{n}_j \mathbf{s}_{ji}}{\pi |\mathbf{s}_{ij}|^4} d\mathbf{s}_i d\mathbf{s}_j \right) \\ &= {}^T\mathbf{n}_i \left(\frac{1}{A_i} \int_{S_i} \int_{S_j} V(\mathbf{s}_i, \mathbf{s}_j) \frac{\mathbf{s}_{ij} {}^T\mathbf{s}_j \mathbf{n}_j}{\pi |\mathbf{s}_{ij}|^4} d\mathbf{s}_i d\mathbf{s}_j \right) \\ &= {}^T\mathbf{n}_i \left(\frac{1}{A_i} \int_{S_i} \int_{S_j} V(\mathbf{s}_i, \mathbf{s}_j) \frac{\mathbf{s}_{ij} {}^T\mathbf{s}_{ji}}{\pi |\mathbf{s}_{ij}|^4} d\mathbf{s}_i d\mathbf{s}_j \mathbf{n}_j \right) \\ &= {}^T\mathbf{n}_i \mathbf{M}_{ij} \mathbf{n}_j \end{aligned}$$

We introduce the symmetric 3×3 *interaction matrix* $\mathbf{M}_{ij}$, defined by two patches S_i and S_j with their respective normal $\mathbf{n}_i$ and $\mathbf{n}_j$ as

$$\mathbf{M}_{ij} = \frac{1}{A_i} \int_{S_i} \int_{S_j} V(\mathbf{s}_i, \mathbf{s}_j) \frac{\mathbf{s}_{ij} {}^T\mathbf{s}_{ji}}{\pi |\mathbf{s}_{ij}|^4} d\mathbf{s}_i d\mathbf{s}_j \quad (\text{B.8})$$

B.3.3 Vector Radiosity equation

With this new definition and introducing the *emissive vector* $\mathbf{E} = E \mathbf{n}$, we rewrite Equation B.5 using Equation B.4:

$$\begin{aligned} {}^T \mathbf{n}_i \mathbf{B}_i &= {}^T \mathbf{n}_i \mathbf{E}_i + r_i \sum_j ({}^T \mathbf{n}_i \mathbf{M}_{ij} \mathbf{n}_j) ({}^T \mathbf{n}_j \mathbf{B}_j) \\ {}^T \mathbf{n}_i \mathbf{B}_i &= {}^T \mathbf{n}_i \left(\mathbf{E}_i + r_i \sum_j ((\mathbf{M}_{ij} \mathbf{n}_j)^T \mathbf{n}_j) \mathbf{B}_j \right) \end{aligned}$$

Introducing the 3×3 symmetric *form-factor matrix* $\mathbf{F}_{ij}$, defined in-between two patches S_i and S_j with their respective normal $\mathbf{n}_i$ and $\mathbf{n}_j$ as

$$\mathbf{F}_{ij} = (\mathbf{M}_{ij} \mathbf{n}_j)^T \mathbf{n}_i \quad (\text{B.9})$$

Similarly to [WHG99], we finally define the *vector radiosity equation* as

$$\mathbf{B}_i = \mathbf{E}_i + r_i \sum_j \mathbf{F}_{ij} \mathbf{B}_j \quad (\text{B.10})$$

B.4 Irradiance Vector for Environment-maps

Bear in mind that an irradiance vector is defined as

$$\mathbf{I}(\mathbf{n}) = \int_{\Omega} L(\boldsymbol{\omega}) \boldsymbol{\omega} h(\mathbf{n}, \boldsymbol{\omega}) d\boldsymbol{\omega}.$$

For a distant lighting field represented as an environment map $\Omega \rightarrow \mathbb{R} : L_{\infty}(\boldsymbol{\omega})$, the incident radiance is thus the product of this distant lighting and a directional visibility function $\Omega \rightarrow \mathbb{R} : V(\boldsymbol{\omega})$:

$$\mathbf{I}(\mathbf{n}) = \int_{\Omega} L_{\infty}(\boldsymbol{\omega}) V(\boldsymbol{\omega}) \boldsymbol{\omega} h(\mathbf{n}, \boldsymbol{\omega}) d\boldsymbol{\omega}.$$

We make the following assumptions:

1. $V(\boldsymbol{\omega}) = 1$ in a cone $\mathcal{C}_{\mathbf{a}, \alpha}$ oriented toward its main axis $\mathbf{a}$ with an angular aperture α
2. the distant lighting is constant in this cone of direction: $\forall \boldsymbol{\omega} \in \mathcal{C}_{\mathbf{a}, \alpha}, L_{\infty}(\boldsymbol{\omega}) = L_{\infty}(\mathcal{C}_{\mathbf{a}, \alpha})$

Thus, the irradiance vector is simplified to be

$$\begin{aligned} \mathbf{I}(\mathbf{n}) &= L_{\infty}(\mathcal{C}_{\mathbf{a}, \alpha}) \int_{\Omega} V(\boldsymbol{\omega}) \boldsymbol{\omega} h(\mathbf{n}, \boldsymbol{\omega}) d\boldsymbol{\omega} \\ &= L_{\infty}(\mathcal{C}_{\mathbf{a}, \alpha}) \mathbf{o}(\mathbf{n}). \end{aligned}$$

We call the vector value of $\Omega \rightarrow \mathbb{R}^3 : \mathbf{o}(\mathbf{n})$ the *directional vector ambient occlusion*. Classical ambient occlusion is simply defined as:

$$\begin{aligned} o &= \frac{1}{\pi} \int_{\Omega} V(\boldsymbol{\omega}) {}^T \mathbf{n} \boldsymbol{\omega} h(\mathbf{n}, \boldsymbol{\omega}) d\boldsymbol{\omega} \\ &= {}^T \mathbf{n} \frac{1}{\pi} \int_{\Omega} V(\boldsymbol{\omega}) \boldsymbol{\omega} h(\mathbf{n}, \boldsymbol{\omega}) d\boldsymbol{\omega} \\ &= {}^T \mathbf{n} \frac{\mathbf{o}(\mathbf{n})}{\pi}. \end{aligned}$$

Based on this assumption of the visibility cone $\mathcal{C}_{\mathbf{a}, \alpha}$, it can be proved that:

1. $\mathbf{o}(\mathbf{n})/|\mathbf{o}(\mathbf{n})| = \mathbf{a}$ if the cone is included in the hemisphere oriented toward $\mathbf{n}$
2. $|\mathbf{o}(\mathbf{n})|$ represent the solid angle of the visible part of the hemisphere. If the cone is included in the hemisphere, $|\mathbf{o}(\mathbf{n})| = \pi(1 - \cos^2(\alpha))$

Generally, there is no restriction in computing the directional ambient occlusion. This provides us with a cone approximation of the visibility by computing the axis direction $\mathbf{a}$ and the angular aperture α as expressed in the previous paragraph. Furthermore, $L_\infty(\mathcal{C}_{\mathbf{a},\alpha})$ can be approximated using a set of prefiltered environment maps (simple mip-mapped environment maps): the level of filtering is controlled by α , and a simple lookup in the selected environment map is performed using the direction $\mathbf{a}$. The final irradiance is thus:

$$I = L_\infty(\mathcal{C}_{\mathbf{o}(\mathbf{n}),|\mathbf{o}(\mathbf{n})|})^T \mathbf{n} \mathbf{o}(\mathbf{n}). \quad (\text{B.11})$$

This approach is very similar to the *accessibility* presented by Pharr and Green [PG04]. Mertens *et al.* [MKC⁺06] do not use a vector representation, but project the visibility function using spherical harmonics. But, our formulation is exact in case of cone-shaped visibility function: it contains the cosine term.

Curriculum Vitae

Xavier Granier

Doctor/Engineer in Computer Science

Born on: 17th of March, 1975

Nationality: French

Address: Domaine Universitaire / 351, cours de la Liberation / 33405 Talence Cedex (France)

Phone: +33 5 40 00 37 95

Fax: +33 5 40 00 38 95

Email: xavier.granier@inria.fr

Web: <http://www.labri.fr/~granier>

Positions

Research Scientist

Since October 2006

INRIA Bordeaux Sud-Ouest - IPARLA project
LaBRI - Bordeaux - France

Junior Research Scientist

October 2003 - October 2006

INRIA Futurs - IPARLA project
LaBRI - Bordeaux - France

Post-doctoral assistant

Decembre 2001 - August 2003

University of British Columbia, Vancouver-Canada
PIMS Post-doctoral Fellowship

PhD student (George Drettakis - Claude Puech)

July 2000-November 2001

REVES/INRIA, Sophia Antipolis-France
iMAGIS/IMAG-INRIA, Grenoble-France

September 1998 - July 2000

iMAGIS/IMAG-INRIA, Grenoble-France

Master student (George Drettakis)

October 1997 - June 1998

iMAGIS/IMAG-INRIA, Grenoble-France

University Diplomas

PhD of Université Joseph Fourier (Grenoble)

"Contrôle automatique de qualité pour l'illumination globale"

Defended the 9th of Novembre 2001 at INRIA Rhône-Alpes with the following jury:

Ms Joëlle Coutaz - President of jury - professor at UJF (Grenoble I)

Mr Peter Shirley - Reviewer - professor at University of Utah (United States)

Mr Bernard Peroche - Reviewer - professor at Université Claude Bernard (Lyon I)

Mr Christophe Schlick - professor at Université de Bordeaux II

Mr George Drettakis - PhD adviser - research scientist at INRIA-Sophia Antipolis

Mr Claude Puech - PhD adviser - professor at UJF (Grenoble I)

Master (DEA) of Institut National Polytechnique de Grenoble (INPG)

Obtained in June 1998

Specialty : Image, Vision and Robotism

Simulation d'Illumination Globale par Méthode de Radiosité avec Mémoire Limitée

Engineer of l'École Nationale Supérieure d'Informatique et de Mathématiques Appliquées de Grenoble (ENSIMAG)

Obtained in June 1998

Specialty : Scientific Computation

Theses: adviser and jury

Co-Advised PhD Theses

- | | |
|--------------------|---|
| Florian Levet: | 2004 - 2006 with Christophe Schlick
<i>Modélisation et Edition d'objets 3D : Du prototype au modèle final</i>
Publications: [LGS06a, LGS06b, LG07, LGS07a, LGS07b, LGS07c]
Ingénieur de recherche à l'INSERM |
| Julien Hadim: | 2005 - 2009 with Christophe Schlick
<i>Étude en vue de la multirésolution de l'apparence</i>
Publications: [HBR ⁺ 07, ZHG08] |
| Romain Pacanowski: | 2005 - 2009 with Christophe Schlick and Pierre Poulin
<i>Modes de représentation pour l'éclairage en synthèse d'images</i>
Publications: [PGS07, PGSP08, PRG ⁺ 08, PRL ⁺ 08, VPB ⁺ 09]
Post-doctorant au CEA CESTAS |
| Romain Vergne: | 2007 - 2010 with Christophe Schlick et Pascal Barla
<i>Techniques expressives pour l'illustration, la présentation et l'exploration de données complexes 3D</i>
Publications: [VBT ⁺ 07, VBGS08a, VBGS08b, VPB ⁺ 09] |

Curriculum Vitae

Juries for PhD Theses

- Jacques Morice, University de Bordeaux 1 - 17th October 2007
Méthode multipôle rapide pour la résolution de l'équation de la radiosité en transfert radiatif
- Pau Estalella Fernández, University of Barcelona - 19th December 2008
Accurate Interactive Specular Reflections and Refractions on Curved Objects

Advised Master Theses

- | | |
|--------------------|--|
| Julien Hadim: | 2003 - 2004
<i>Apparence multirésolution</i> |
| Romain Pacanowski: | 2004 - 2005
<i>Reconstruction de la fonction d'éclairage</i> |
| Romain Vergne: | 2005 - 2006
<i>Stylisation d'objets éclairés par des cartes d'environnement de grandes dynamiques</i> |

Involvements in Scientific Community

Publications

I have been member of the following conference **program committees**:

- Graphics Interface: 2004-2005, 2008-2009
- Eurographics Sketch-Based Interfaces and Modeling Graphics: 2008-2009

I have been involved in the **reviewing** process of the following journals, books and conferences:

- Morgan Kaufmann: book - 2008
- Eurographics Computer Graphics Forum: journal - 2002-2003, 2005, 2007-2009
- Parallel Computing - Elsevier: journal - 2005
- ACM Journal of Graphic Tools: journal - 2002
- IEEE Computer Graphics and Applications: journal - 2008
- ACM SIGGRAPH: journal/conference - 2002-2006, 2008
- Eurographics Symposium on Rendering: journal/conference - 2003, 2005-2008
- Eurographics (Annual Conference): journal/conference - 2002, 2004-2006, 2008
- PDP (Euromicro International Conference on Parallel, Distributed and network-based Processing): conference - 2008
- ACM Graphite (International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia): conference - 2007
- IET VIE (Visual Information Engineering): conferences - 2006-2007
- Graphics Interface: conference - 2002-2005, 2008-2009
- IEEE Visualization: conference - 2005, 2007

Grants

I have been in charge of or participated to the **coordination** of the following grants:

- **Associated Team LIGHT:** "Lab for Interactive Graphics on Handheld and Tabletop displays"
Grant: INRIA-DREI
Dates: 2004-2008
Partners: IMAGER Lab - University of British Columbia - Vancouver - Canada
Publications: [BHGS06, GGHS06]
 LIGHT stands for "Laboratory for Interactive Graphics on Handheld and Tabletop Displays". Our goal is to investigate the different possibilities for the acquisition, the rendering and the visualization of specific or adapted representation of mobile and connected devices. We want also to develop some new interaction techniques for such devices.
- **MIRO**
Grant: INRIA ARC
Dates: 2005-2006
Partners: INRIA (IPARLA, ARTIS) + ISTI-Pise, AUSONIUS-Bordeaux, LSC-Bordeaux, DIGISENS-Annecy
Publications: [QTG⁺06]
 MIRO stands for "Methods for the legible and Interactive Rendering Of Complex data". In this project, we are interested in using NPR for the interactive and legible rendering of complex 3D scenes. Many examples of scientific visualization prove that photorealism does not always offer meaningful images. To address the legibility question we will rely (i) on the skills and experience of the IPARLA and ARTIS teams; (ii) on the knowledge of specific users: archeologists, museum curators, industrial users at Digisens (the startup specialized in CTscan reconstruction and dental chirurgy); (iii) on the skills of the Bordeaux lab of cognitive psychology that will help us how to validate our new methods.
- **France-Télécom:** "Modeling and rendering of non-photorealistic urban environments and digital territories."
Dates: 2005-2006
Publications: [QTG⁺06]
 This project concerns the modeling and rendering of non-photorealistic urban environments and digital territories. The target platforms for visualization are small devices. We will propose a whole graphic pipeline (data modeling, transmission and rendering) that is adapted to small devices constraints. We will implement a client-server application where the server owns a huge 3D database which is rendered on the client (a small device). As an alternative to photorealistic rendering and texture based rendering which generate large models, we will explore non-photorealistic techniques as line based rendering. Point-based rendering will also be considered. Some multi-resolution models of cities or territories (efficiently transmitted and rendered) will be constructed with those techniques.

I have participated and I participate to the following grants:

- **SHOW**
Grant: ARA "Masse de données" (Research National Agency)
Dates: 2003-2006
Partners: ARTIS et GRAVIR-IMAG (Inria Rhones-Alpes), ISA-ALICE Loria (Inria Lorraine), REVES (Inria Sophia-Antipolis)
Publications: [BHGS06]
 3D numerical data are often large and un-structured. This is mainly due to the required acquisition or automatic transformation processes. This project aims to re-create a structure, or a

hierarchy, in order to be allowed to reconstruct and visualize them interactively.

- **NatSim**

Grant: ARA "Masse de données" (Research National Agency)

Dates: 2005 - 2008

Partners: IRIT (Toulouse 3), EVASION (Inria Rhones-Alpes), Virtual Plants (INRIA joined with CIRAD and INRA), LIAMA (Beijing)

Publications: [HBR⁺07, PRG⁺08]

This project deals with natural simulations (vegetation, watercourses, clouds). It aims to adapt this huge amount of heterogeneous data in terms of data structures, techniques and algorithms, in a unified framework to both to the content and navigation context (from mobile phones to display walls). The final presentation of NatSim project was done at ANR on December 2008. This project permits to exchange expertise with other french labs and permits to create new collaborations. The main collaborations of IPARLA members into the NatSIM project took place into 4 workpackages : Edition and manipulation of big amount of data, Rendering, Animation and Streaming.

- **Animaré**

Grant: ANR Jeune Chercheur (Research National Agency)

Dates: 2009 - 2012

Partners: ARTIS (INRIA Rhone Alpes)

Publications: [VPB⁺09]

Expressive Rendering is a recent branch of Computer Graphics that offers promising novel styles, and is increasingly used in many application domains such as video games or movie production. At the present time, only expert artists are able to create compelling animations, and still, this is an extremely time-consuming process, with many constraints that strongly limit creativity. The reason is that current models are not sophisticated enough to provide intuitive manipulations and versatile styles. The motivation behind this project is to overcome these limitations both for 2D and 3D animation systems.

- **Associated Team Bird:** "Interactions entre les mondes Réels et Virtuels / Interactions between Real and Virtual Worlds"

Grant: INRIA-DREI

Dates: 2008-2011

Partners: Bunraku - IRISA - Rennes and State key Lab of CAD&CG - Zhejiang University - Hangzhou - China

Publications: [ZHG08]

The main purpose of this collaboration is to provide new tools for managing the interaction between real and virtual worlds. We first want to ease the interaction between real users and virtual worlds during modeling and collaborative tasks. Concerning generation of virtual worlds, we will focus not on fully automatic solutions, but on semi-automatic ones that will take into account human decisions. This integration of the user is essential to provide intuitive processes and better immersion. Based on the different interfaces between virtual and real world (from a simple stylus to a set of cameras), we have to capture accurately the motions, the gestures, and to interpret the intentions of humans, in order to correctly integrate these actions and intentions.

- **SeARCH:** Semi-automatique Acquisition and Reconstruction of Cultural Heritage

Grant: ANR - ContInt call Dates: 2009 - 2010

Partners: Ausonius (CNRS - Université de Bordeaux), CEAlex (CNRS - Alexandrie - Egypte), ESTIA (Bidard - France)

The SeARCH project is particularly motivated by a concrete archeological context: one of the

partners is the Centre d'Études Alexandrines (CEAlex, USR 3134) that works on the reconstruction of the lighthouse of Alexandria and its surrounding statues. Most of the fragments of the lighthouse and the statues are underwater. Some of the fragments, especially from the statues, have already been lifted to the surface. The SeARCH project strives to develop semi-automatic techniques for the virtual reassembly of 3D objects. The first involved step is the digital acquisition of the fragments, on-site, and under aggravated circumstances, combined with some post-processing steps of the acquired fragments. The second step is the reassembly of the fragments that should not only be as automatic as possible, but should also allow taking into account the long-year work experience of the cultural heritage professionals by new efficient interaction and visualization techniques.

I have participated to the **evaluation of grant proposals** for:

- **Regional Grant:** dossier de recherche régionale de l'Université de Reims Champagne-Ardenne - 2007
- **Regional Grant:** Agence pour la Recherche et l'Innovation CARINNA -2008
- **Agence National de la Recherche:** Program "Masse de Données et COonnaissances" (MDCO) - 2007

Others

- **Recruiting jury** for junior research scientist (CR2) at INRIA Futurs Lille: 2008
- **Recruiting committee** for University of Bordeaux 1, ENSEIRB and, IUT of Bordeaux: 2004-2008
- **Europe Correspondent** for INRIA Futurs Bordeaux: 2007-2008
- LaBRI-Hebdo **chief editor:** 2006 - 2008

Publications

In the following, **journals** are in red (7 publications in reference journals), **books and book chapters** in green (2 chapters) and, **conferences** in blue (16 international conferences with reviewing committee).

Selection

- [PRG⁺08] Romain Pacanowski, Mickaël Raynaud, **Xavier Granier**, Patrick Reuter, Christophe Schlick, and Pierre Poulin. Efficient Streaming of 3D Scenes with Complex Geometry and Complex Lighting. In *Web3D '08: Proc. international symposium on 3D web technology*, pages 11–17. ACM, 2008.
- [VBGS08a] Romain Vergne, Pascal Barla, **Xavier Granier**, and Christophe Schlick. Apparent relief: a shape descriptor for stylized shading. In *NPAR '08: Proc. international symposium on Non-photorealistic animation and rendering*, pages 23–29. ACM, 2008.
- [BHGS06] Tamy Boubekeur, Wolfgang Heidrich, **Xavier Granier**, and Christophe Schlick. Volume-Surface Trees. *Comp. Graph. Forum (Proc. EUROGRAPHICS 2006)*, 25(3):399–406, 2006, **Best Paper and Best Student Paper awards**.
- [GD04b] **Xavier Granier** and George Drettakis. A Final Reconstruction Framework for an Unified Global Illumination Algorithm. *ACM Trans. Graph.*, 23(2):163–189, 2004.
- [GGHS03a] Michael Goesele, **Xavier Granier**, Wolfgang Heidrich, and Hans-Peter Seidel. Accurate light source acquisition and rendering. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 22(3):621–630, 2003.

Whole List

Reference (22)

- [VPB⁺09] Romain Vergne, Romain Pacanowski, Pascal Barla, **Xavier Granier**, and Christophe Schlick. Light warping for enhanced surface depiction. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 28(3), 2009. to be published.
- [PRG⁺08] Romain Pacanowski, Mickaël Raynaud, **Xavier Granier**, Patrick Reuter, Christophe Schlick, and Pierre Poulin. Efficient Streaming of 3D Scenes with Complex Geometry and Complex Lighting. In *Web3D '08: Proc. international symposium on 3D web technology*, pages 11–17. ACM, 2008.
- [PGSP08] Romain Pacanowski, **Xavier Granier**, Christophe Schlick, and Pierre Poulin. Sketch and Paint-based Interface for Highlight Modeling. In *SBIM 2008: EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling*, pages 11–17. Eurographics, 2008.
- [VBGS08a] Romain Vergne, Pascal Barla, **Xavier Granier**, and Christophe Schlick. Apparent relief: a shape descriptor for stylized shading. In *NPAR '08: Proc. international symposium on Non-photorealistic animation and rendering*, pages 23–29. ACM, 2008.
- [GPP07a] **Xavier Granier**, Mathias Paulin, and Bernard Péroche. *Informatique graphique et rendu*, chapter 5 - Couleur; Modèles locaux d'éclairément; Ombres portées. Hermes Publishing, 2007.

- [GPP07b] **Xavier Granier**, Mathias Paulin, and Bernard Péroche. *Informatique graphique et rendu*, chapter 7 - Rendu réaliste : algorithmes pour l'éclairage global. Hermes Publishing, 2007.
- [HBR⁺07] Julien Hadim, Tamy Boubekur, Mickaël Raynaud, **Xavier Granier**, and Christophe Schlick. On-the-fly appearance quantization on the gpu for 3d broadcasting. In *Web3D '07: Proc. international conference on 3D web technology*, pages 45–51. ACM, 2007.
- [LG07] Florian Levet and **Xavier Granier**. Improved Skeleton Extraction and Surface Generation for Sketch-based Modeling. In *Proc. Graphics Interface 2007*, pages 27–33. A.K. Peters, 2007.
- [BHGS06] Tamy Boubekur, Wolfgang Heidrich, **Xavier Granier**, and Christophe Schlick. Volume-Surface Trees. *Comp. Graph. Forum (Proc. EUROGRAPHICS 2006)*, 25(3):399–406, 2006, **Best Paper and Best Student Paper awards**.
- [LGS06b] Florian Levet, **Xavier Granier**, and Christophe Schlick. Fast sampling of implicit surfaces by particle systems. In *SMI '06: Proc. Shape Modeling International 2006*, page 39. IEEE Computer Society, 2006.
- [QTG⁺06] Jean-Charles Quillet, Gwenola Thomas, **Xavier Granier**, Pascal Guitton, and Jean-Eudes Marvie. Using Expressive Rendering for Remote Visualization of Large City Models. In *Web3D '06: Proc. international conference on 3D web technology*, pages 27–35. ACM, 2006.
- [GD04b] **Xavier Granier** and George Drettakis. A Final Reconstruction Framework for an Unified Global Illumination Algorithm. *ACM Trans. Graph.*, 23(2):163–189, 2004.
- [GGHS03a] Michael Goesele, **Xavier Granier**, Wolfgang Heidrich, and Hans-Peter Seidel. Accurate light source acquisition and rendering. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 22(3):621–630, 2003.
- [GGHS03b] **Xavier Granier**, Michael Goesele, Wolfgang Heidrich, and Hans-Peter Seidel. Interactive Visualization of Complex Real-World Light Sources. In *PG '03: Proc. Pacific Conference on Computer Graphics and Applications*, pages 59–66. IEEE Computer Society, 2003.
- [GH03] **Xavier Granier** and Wolfgang Heidrich. A Simple Layered RGB BRDF Model. *Graphical Models*, 65(4):169–257, 2003. Extended version of [GH02].
- [GH02] **Xavier Granier** and Wolfgang Heidrich. A Simple Layered RGB BRDF Model. In *PG '02: Proc. Pacific Conference on Computer Graphics and Applications*, page 30. IEEE Computer Society, 2002.
- [GD01] **Xavier Granier** and George Drettakis. Incremental Updates for Rapid Glossy Global Illumination. *Comp. Graph. Forum (Proc. EUROGRAPHICS 2001)*, 20(3):267–277, 2001.
- [GDW00] **Xavier Granier**, George Drettakis, and Bruce Walter. Fast Global Illumination Including Specular Effects. In *Proc. EUROGRAPHICS Workshop on Rendering 2000*, pages 47 – 59. Springer-Verlag GmbH, 2000.
- [SSG⁺00] Marc Stamminger, Annette Scheel, **Xavier Granier**, Frédéric Perez-Carzorla, George Drettakis, and François Sillion. Efficient Glossy Global Illumination with Interactive Viewing. *Comp. Graph. Forum*, 19(1):13–25, 2000. Extended version of [SSG⁺99].
- [GD99] **Xavier Granier** and George Drettakis. Controlling memory consumption of hierarchical radiosity with clustering. In *Proc. Graphics Interface 99*, pages 58–65. Morgan Kaufmann Publishers, 1999.

- [LFD⁺99] Céline Loscos, Marie-Claude Frasson, George Drettakis, Bruce Walter, **Xavier Granier**, and Pierre Poulin. Interactive virtual relighting and remodeling of real scenes. In *Proc. EUROGRAPHICS Workshop on Rendering '99*, pages 235–246. Springer-Verlag GmbH, 1999.
- [SSG⁺99] Marc Stamminger, Annette Scheel, **Xavier Granier**, Frédéric Perez-Carzorla, George Drettakis, and François Sillion. Efficient Glossy Global Illumination with Interactive Viewing. In *Proc. Graphics Interface 99*, pages 50–57. Morgan Kaufmann Publishers, 1999.

Others With Reviewing Committee (6)

- [PRL⁺08] Romain Pacanowski, Mickaël Raynaud, Julien Lacoste, **Xavier Granier**, Patrick Reuter, Christophe Schlick, and Pierre Poulin. Compact structures for interactive global illumination on large cultural objects. *International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST 2008): Shorts and Projects*, 2008.
- [VBGS08b] Romain Vergne, Pascal Barla, **Xavier Granier**, and Christophe Schlick. Shading with apparent relief. In *SIGGRAPH '08: ACM SIGGRAPH 2008 talks*. ACM, 2008.
- [ZHG08] Hongxin Zhang, Julien Hadim, and **Xavier Granier**. Using the CAT for 3D Sketching in front of Large Displays. In *International Symposium on Smart Graphics, Lecture Notes on Computer Science*. Springer-Verlag GmbH, 2008.
- [LGS07b] Florian Levet, **Xavier Granier**, and Christophe Schlick. Multi-View Sketch-based FreeForm Modeling. In *International Symposium on Smart Graphics, Lecture Notes on Computer Science*. Springer-Verlag GmbH, 2007.
- [LGS06a] Florian Levet, **Xavier Granier**, and Christophe Schlick. 3D Sketching with profile curves. In *International Symposium on Smart Graphics, Lecture Notes on Computer Science*. Springer-Verlag GmbH, 2006.
- [KGB05] Bertrand Kerautret, **Xavier Granier**, and Achille Braquelaire. Intuitive Shape Modeling by Shading Design. In *International Symposium on Smart Graphics*, volume 3638 of *Lecture Notes in Computer Science*, pages 163–174. Springer-Verlag GmbH, 2005.

Others

- [PGSP09] Romain Pacanowski, **Xavier Granier**, Christophe Schlick, and Pierre Poulin. Volumetric Vector-Based Representation for Indirect Illumination Caching. Research Report RR-6983, INRIA, 2009.
- [LGS07a] Florian Levet, **Xavier Granier**, and Christophe Schlick. MarchingParticles: Fast Generation of Particles for the Sampling of Implicit Surfaces. *Computer Graphics & Geometry*, 9(1):18–49, 2007.
- [LGS07c] Florian Levet, **Xavier Granier**, and Christophe Schlick. Triangulation of uniform particle systems: its application to the implicit surface texturing. In *WSCG (Winter School of Computer Graphics)*, 2007.
- [PGS07] Romain Pacanowski, **Xavier Granier**, and Christophe Schlick. Gestion de la complexité géométrique dans le calcul d'éclairage pour la présentation publique de scènes archéologiques complexes. In *Virtual Retrospect 2007 : Archéologie et Réalité Virtuelle*. Ausonius, 2007.

- [VBT⁺07] Romain Vergne, Adrien Bousseau, Joëlle Thollot, David Vanderhaeghe, Pascal Barla, and **Xavier Granier**. Utilisation du rendu expressif pour l'illustration et l'exploration de données archéologiques. In *Virtual Retrospect 2007 : Archéologie et Réalité Virtuelle*. Ausonius, 2007.
- [GGHS06] **Xavier Granier**, Michael Goesele, Wolfgang Heidrich, and Hans-Peter Seidel. Optical filtering for near field photometry with high order basis. Research Report RR-6000, INRIA, 2006.
- [GD04a] **Xavier Granier** and Cyrille Domez. Control on Color Reflection Behavior. Research Report RR-5227, INRIA, 2004.
- [Gra01] **Xavier Granier**. *Contrôle Automatique de Qualité pour l'Illumination Globale*. PhD thesis, Université Joseph Fourier (Grenoble 1), 2001.

Bibliography

- [AA03] Anders Adamson and Marc Alexa. Ray Tracing Point Set Surfaces. In *SMI '03: Proc. Shape Modeling International 2003*, page 272. IEEE Computer Society, 2003.
- [ABCG04] Anca Alexe, Loïc Barthe, Marie-Paule Cani, and Véronique Gaildrat. Interactive modelling from sketches using spherical implicit functions. In *AFRIGRAPH '04: Proc. conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, pages 25–34. ACM, 2004.
- [ABCG05] Anca Alexe, Loïc Barthe, Marie-Paule Cani, and Véronique Gaildrat. Shape Modeling by Sketching using Convolution Surfaces. In *PG '05: Proc. Pacific Conference on Computer Graphics and Applications*, Short paper, 2005.
- [ACSD⁺03] Pierre Alliez, David Cohen-Steiner, Olivier Devillers, Bruno Lévy, and Mathieu Desbrun. Anisotropic Polygonal Remeshing. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 22(3):485–493, 2003.
- [AFO05] Okan Arikan, David A. Forsyth, and James F. O’Brien. Fast and Detailed Approximate Global Illumination by Irradiance Decomposition. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 24(3):1108 – 1114, 2005.
- [AJ03] Bruno Rodrigues De Araujo and Joaquim A. Jorge. Blobmaker: Free-form modelling with variational implicit surfaces. In *Proc. “12th Encontro Português de Computação Gráfica”*, pages 17–26, 2003.
- [AKDS04] Thomas Annen, Jan Kautz, Frédo Durand, and Hans-Peter Seidel. Spherical Harmonic Gradients for Mid-Range Illumination. In *Proc. EUROGRAPHICS Symposium on Rendering 2004*, pages 331–336. EUROGRAPHICS, 2004.
- [APS00] Michael Ashikhmin, Simon Premoze, and Peter Shirley. A microfacet-based BRDF generator. In *Proc. SIGGRAPH '00*, annual conference on Computer graphics and interactive techniques, pages 65–74. ACM/Addison-Wesley Publishing Co., 2000.
- [Arv94] James R. Arvo. The irradiance Jacobian for partially occluded polyhedral sources. In *Proc. SIGGRAPH '94*, annual conference on Computer graphics and interactive techniques, pages 343–350. ACM, 1994.
- [Arv95] James R. Arvo. Applications of irradiance tensors to the simulation of non-Lambertian phenomena. In *Proc. SIGGRAPH '95*, annual conference on Computer graphics and interactive techniques, pages 335–342. ACM, 1995.
- [Ash94] Ian Ashdown. *Radiosity: A Programmer’s Perspective*. John Wiley & Sons, 1994.
- [AWB06] Ken-Ichi Anjyo, Shuhei Wemler, and William Baxter. Tweakable light and shade for cartoon animation. In *Proc. international symposium on Non-Photorealistic Animation and Rendering*, pages 133–139. ACM, 2006.
- [BAERD08] Aner Ben-Artzi, Kevin Egan, Ravi Ramamoorthi, and Frédo Durand. A precomputed polynomial representation for interactive BRDF editing with global illumination. *ACM Trans. Graph.*, 27(2):1–13, 2008.
- [BAOR06] Aner Ben-Artzi, Ryan Overbeck, and Ravi Ramamoorthi. Real-time BRDF editing in complex lighting. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 25(3):945–954, 2006.
- [BCD01] David Bourguignon, Marie-Paule Cani, and George Drettakis. Drawing for Illustration and Annotation in 3D. *Comp. Graph. Forum (Proc. EUROGRAPHICS 2001)*, 20(3):114–122, 2001.
- [BDM09] Steven Bergner, Mark S. Drew, and Torsten Möller. A tool to create illuminant and reflectance spectra for light-driven graphics and visualization. *ACM Trans. Graph.*, 28(1):1–11, 2009.
- [BFKB99] Ravin Balakrishnan, George Fitzmaurice, Gordon Kurtenbach, and William Buxton. Digital tape drawing. In *UIST '99: Proc. symposium on User interface software and technology*, pages 161–169. ACM, 1999.
- [BHGS06] Tamy Boubekeur, Wolfgang Heidrich, Xavier Granier, and Christophe Schlick. Volume-Surface Trees. *Comp. Graph. Forum (Proc. EUROGRAPHICS 2006)*, 25(3):399–406, 2006.
- [Bli77] James F. Blinn. Models of light reflection for computer synthesized pictures. In *Proc. SIGGRAPH '77*, pages 192–198. ACM, 1977.
- [Bor91] Carlos F. Borges. A Trichromatic Approximation Method for Surface Illumination. *J. Optical Society of America*, 8(8):1319–1323, 1991.

- [BPCB08] Adrien Bernhardt, Adeline Pihuit, Marie-Paule Cani, and Loïc Barthe. Matisse: Painting 2D regions for Modeling Free-Form Shapes. In *SBIM 2008: EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling*, pages 57–64, 2008.
- [BTM06] Pascal Barla, Joëlle Thollot, and Lee Markosian. X-Toon: An extended toon shader. In *NPAP '06: Proc. international symposium on Non-photorealistic animation and rendering*, pages 127–132. ACM, 2006.
- [BTS05] Pascal Barla, Joëlle Thollot, and François Sillion. Geometric Clustering for Line Drawing Simplification. In *Proc. EUROGRAPHICS Symposium on Rendering 2005*. EUROGRAPHICS, 2005.
- [CAH00] François Cuny, Laurent Alonso, and Nicolas Holzschuch. A novel approach makes higher order wavelets really efficient for radiosity. *Comp. Graph. Forum*, 19(3):C99–C108, 2000.
- [CB04] Per H. Christensen and Dana Batali. An Irradiance Atlas for Global Illumination in Complex Production Scenes. In *Proc. EUROGRAPHICS Symposium on Rendering 2004*, pages 133–141, 2004.
- [CBS96] Benoît Crespín, Carole Blanc, and Christophe Schlick. Implicit Sweep Objects. *Comp. Graph. Forum (Proc. EUROGRAPHICS '96)*, 15(3):165–174, 1996.
- [CHL04] Greg Coombe, Mark J. Harris, and Anselmo Lastra. Radiosity on graphics hardware. In *Proc. Graphics Interface 2004*, pages 161–168. Canadian Human-Computer Communications Society, A.K. Peters Ltd, 2004.
- [Chr99] Per H. Christensen. Faster Global Photon Map Global Illumination. *J. of Graphics Tools*, 4(3):1–10, 1999.
- [CIE] Commission Internationale de l’Eclairage. <http://www.cie.co.at/>.
- [CMS87] Brian Cabral, Nelson Max, and Rebecca Springmeyer. Bidirectional reflection functions from surface bump maps. In *Proc. SIGGRAPH '87*, annual conference on Computer graphics and interactive techniques, pages 273–281. ACM, 1987.
- [CMZ⁺99] Jonathan M. Cohen, Lee Markosian, Robert C. Zeleznik, John F. Hughes, and Ronen Barzel. An interface for sketching 3D curves. In *SI3D '99: Proc. symposium on interactive 3D graphics*, pages 17–21. ACM, 1999.
- [Cou06] Michel Couprie. Note on fifteen 2d parallel thinning algorithms. Technical Report GM 2006-01, Institut Gaspard Monge, 2006.
- [CPK06] Mark Colbert, Sumanta N. Pattanaik, and Jaroslav Křivánek. BRDF-Shop: Creating Physically Correct Bidirectional Reflectance Distribution Functions. *IEEE Comp. Graph. Appl.*, 26(1):30–36, 2006.
- [CRMT91] Shenchang Eric Chen, Holly E. Rushmeier, Gavin Miller, and Douglass Turner. A progressive multi-pass method for global illumination. In *Proc. SIGGRAPH '91*, annual conference on Computer graphics and interactive techniques, pages 165–174. ACM, 1991.
- [CSSD94] Per H. Christensen, Eric J. Stollnitz, David H. Salesin, and Tony D. DeRose. Wavelet Radiance. In *Proc. EUROGRAPHICS Workshop on Rendering '94*, pages 287–302. Springer-Verlag GmbH, 1994.
- [CSSJ05] Joseph Jacob Cherlin, Faramarz Samavati, Mario Costa Sousa, and Joaquim A. Jorge. Sketch-based modeling with few strokes. In *SCCG '05: Proc. spring conference on Computer graphics*, pages 137–145, 2005.
- [CT82] Robert L. Cook and Kenneth E. Torrance. A Reflectance Model for Computer Graphics. *ACM Trans. Graph.*, 1(1):7–24, 1982.
- [DB00] Reynald Dumont and Kadi Bouatouch. Combining Hierarchical Radiosity and LODs. In *Proc. IASTED International Conference on Computer Graphics and Imaging*, 2000.
- [DBB06] Philip Dutré, Kavita Bala, and Philippe Bekaert. *Advanced Global Illumination (Second Edition)*. A. K. Peters, Ltd., 2006.
- [DBG99] Reynald Dumont, Kadi Bouatouch, and Philippe Gosselin. A Progressive Algorithm for Three Point Transport. *Comp. Graph. Forum*, 18(1):41–56, 1999.
- [DBMS02] Kirill Dmitriev, Stefan Bräbec, Karol Myszkowski, and Hans-Peter Seidel. Interactive global illumination using selective photon tracing. In *Proc. EUROGRAPHICS workshop on Rendering*, pages 25–36. EUROGRAPHICS, 2002.
- [Dec96] Philippe Decaudin. Cartoon Looking Rendering of 3D Scenes. Research Report 2919, INRIA, 1996.
- [DF00] Mark S. Drew and Graham D. Finlayson. Spectral Sharpening with Positivity. *J. Optical Society of America*, 17(8):1361–1370, 2000.
- [DF02] Mark S. Drew and Graham D. Finlayson. Multispectral Processing Without Spectra. Technical Report SFU-CMPT-TR2002-02, School Of Computing Science - Simon Fraser University, 2002.
- [DFR04] Doug DeCarlo, Adam Finkelstein, and Szymon Rusinkiewicz. Interactive Rendering of Suggestive Contours with Temporal Coherence. In *NPAP '04: Proc. international symposium on Non-photorealistic animation and rendering*, pages 15–24. ACM, 2004.
- [DFRS03] Doug DeCarlo, Adam Finkelstein, Szymon Rusinkiewicz, and Anthony Santella. Suggestive contours for conveying shape. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 22(3):848–855, 2003.
- [DHS⁺05] Frédo Durand, Nicolas Holzschuch, Cyril Soler, Eric Chan, and François Sillion. A Frequency Analysis of Light Transport. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 24(3), 2005.
- [DHT⁺00] Paul E. Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. In *Proc. SIGGRAPH '00*, annual conference on Computer graphics and interactive techniques, pages 145–156. ACM/Addison-Wesley Publishing Co., 2000.

- [DM97] Paul E. Debevec and Jitendra Malik. Recovering High Dynamic Range Radiance Maps from Photographs. In *Proc. SIGGRAPH '97*, annual conference on Computer graphics and interactive techniques, pages 369–378. ACM, 1997.
- [DR07] Doug DeCarlo and Szymon Rusinkiewicz. Highlight lines for conveying shape. In *NPAR '07: Proc. international symposium on Non-photorealistic animation and rendering*, pages 63–70. ACM, 2007.
- [DS97] George Drettakis and François X. Sillion. Interactive update of global illumination using a line-space hierarchy. In *Proc. SIGGRAPH '97*, annual conference on Computer graphics and interactive techniques, pages 57–64. ACM/Addison-Wesley Publishing Co., 1997.
- [DSDD07] Carsten Dachsbacher, Marc Stamminger, George Drettakis, and Frédo Durand. Implicit Visibility and Anti-radiance for Interactive Global Illumination. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 26(3), 2007.
- [DvNK97] Kristin J. Dana, Bram van Ginneken, Shree K. Nayar, and Jan J. Koenderink. Reflectance and texture of real-world surfaces. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 151–157, 1997.
- [DXS⁺07] Julie Dorsey, Songhua Xu, Gabe Smedresman, Holly E. Rushmeier, and Leonard McMillan. The Mental Canvas: A Tool for Conceptual Architectural Design and Analysis. In *PG '07: Proc. Pacific Conference on Computer Graphics and Applications*, pages 201–210. IEEE Computer Society, 2007.
- [DYN04] Yoshinori Dobashi, Tsuyoshi Yamamoto, and Tomoyuki Nishita. Radiosity for Point-Sampled Geometry. In *PG '04: Proc. Pacific Conference on Computer Graphics and Applications*, pages 152–159. IEEE Computer Society, 2004.
- [EBJ⁺06] Dave Edwards, Solomon Boulos, Jared Johnson, Peter Shirley, Michael Ashikhmin, Michael Stark, and Chris Wyman. The halfway vector disk for BRDF modeling. *ACM Trans. Graph.*, 25(1):1–18, 2006.
- [EHBE97] Lynn Eggi, Ching-Yao Hsu, Beat D. Bruderlin, and Gershon Elber. Inferring 3D Models from Freehand Sketches and Constraints. *Computer-Aided Design*, 29(2):101–112, 1997.
- [FDF94] Graham D. Finlayson, Mark S. Drew, and Brian V. Funt. Spectral Sharpening: sensor transformations for improved constancy. *J. Optical Society of America*, 11(5):1553–1563, 1994.
- [FFJ04] Manuel J. Fonseca, Alfredo Ferreira, and Joaquim A. Jorge. Towards 3D Modeling using Sketches and Retrieval. In *SBIM 2004: EUROGRAPHICS Workshop on Sketch-Based Interface*, 2004.
- [GBK⁺02] Tovi Grossman, Ravin Balakrishnan, Gordon Kurtenbach, George Fitzmaurice, Azam Khan, and William Buxton. Creating principal 3D curves with digital tape drawing. In *CHI '02: Proc. SIGCHI conference on Human factors in computing systems*, pages 121–128. ACM, 2002.
- [GD99] Xavier Granier and George Drettakis. Controlling Memory Consumption of Hierarchical Radiosity with Clustering. In *Proc. Graphics Interface 99*, pages 58–65. Morgan Kaufmann Publishers, 1999.
- [GD01] Xavier Granier and George Drettakis. Incremental Updates for Rapid Glossy Global Illumination. *Comp. Graph. Forum (Proc. EUROGRAPHICS 2001)*, 20(3):267–277, 2001.
- [GD04a] Xavier Granier and Cyrille Domez. Control on Color Reflection Behavior. Research Report RR-5227, INRIA, 2004.
- [GD04b] Xavier Granier and George Drettakis. A Final Reconstruction Framework for an Unified Global Illumination Algorithm. *ACM Trans. Graph.*, 23(2):163–189, 2004.
- [GDW00] Xavier Granier, George Drettakis, and Bruce Walter. Fast Global Illumination Including Specular Effects. In *Proc. EUROGRAPHICS Workshop on Rendering 2000*, pages 47 – 59. Springer-Verlag GmbH, 2000.
- [GG07] Gaël Guennebaud and Markus Gross. Algebraic point set surfaces. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 26(3):23, 2007.
- [GGHS03a] Michael Goesele, Xavier Granier, Wolfgang Heidrich, and Hans-Peter Seidel. Accurate Light Source Acquisition and Rendering. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 22(3):621–630, 2003.
- [GGHS03b] Xavier Granier, Michael Goesele, Wolfgang Heidrich, and Hans-Peter Seidel. Interactive Visualization of Complex Real-World Light Sources. In *PG '03: Proc. Pacific Conference on Computer Graphics and Applications*, pages 59–66. IEEE Computer Society, 2003.
- [GGHS06] Xavier Granier, Michael Goesele, Wolfgang Heidrich, and Hans-Peter Seidel. Optical Filtering for Near Field Photometry with High Order Basis. Research Report RR-6000, INRIA, 2006.
- [GH96] Simon Gibson and Roger J. Hubbard. Efficient Hierarchical Refinement and Clustering for Radiosity in Complex Environments. *Comp. Graph. Forum*, 15(5):297–310, 1996.
- [GH98] Cindy Grimm and John F. Hughes. Implicit Generalized Cylinders using Profile Curves. In *Implicit Surfaces*, pages 33–41, 1998. Creating sweep surfaces using sketching.
- [GH02] Xavier Granier and Wolfgang Heidrich. A simple layered rgb brdf model. In *PG '02: Proc. Pacific Conference on Computer Graphics and Applications*, page 30. IEEE Computer Society, 2002.
- [GH03] Xavier Granier and Wolfgang Heidrich. A Simple Layered RGB BRDF Model. *Graphical Models*, 65(4):169–257, 2003. Extended version of [GH02].
- [GKBP05] Pascal Gautron, Jaroslav Krivánek, Kadi Bouatouch, and Sumanta N. Pattanaik. Radiance cache splatting: a GPU-friendly global illumination algorithm. In *Proc. EUROGRAPHICS Symposium on Rendering 2005*, pages 55–64. EUROGRAPHICS, 2005.

- [GKMD06] Paul Green, Jan Kautz, Wojciech Matusik, and Frédo Durand. View-dependent precomputed light transport using nonlinear Gaussian function approximations. In *I3D '06: Proc. symposium on Interactive 3D graphics and games*, pages 7–14. ACM, 2006.
- [GKPB04] Pascal Gautron, Jaroslav Křivánek, Sumanta N. Pattanaik, and Kadi Bouatouch. A Novel Hemispherical Basis for Accurate and Efficient Rendering. In *Proc. EUROGRAPHICS Symposium on Rendering 2004*, pages 321–330. EUROGRAPHICS, 2004.
- [Gla89a] Andrew S. Glassner. *An Introduction to Ray tracing*. Morgan Kaufmann Publishers, 1989.
- [Gla89b] Andrew S. Glassner. How to derive a Spectrum From a RGB triplet. *IEEE Comp. Graph. Appl.*, 9(4):95–99, 1989.
- [GPP07a] Xavier Granier, Mathias Paulin, and Bernard Péroche. *Informatique graphique et rendu*, chapter 5 - Couleur; Modèles locaux d'éclairément; Ombres portées. Hermes Publishing, 2007.
- [GPP07b] Xavier Granier, Mathias Paulin, and Bernard Péroche. *Informatique graphique et rendu*, chapter 7 - Rendu réaliste : algorithmes pour l'éclairément global. Hermes Publishing, 2007.
- [Gra01] Xavier Granier. *Contrôle Automatique de Qualité pour l'Illumination Globale*. PhD thesis, Université Joseph Fourier (Grenoble 1), 2001.
- [GSA03] Enrico Gobbetti, Leonardo Spanò, and Marco Agus. Hierarchical higher order face cluster radiosity for global illumination walkthroughs of complex non-diffuse environments. *Comp. Graph. Forum (Proc. EUROGRAPHICS)*, 22(3):563–572, 2003.
- [GSCH93] Steven J. Gortler, Peter Schröder, Michael F. Cohen, and Pat Hanrahan. Wavelet radiosity. In *Proc. SIGGRAPH '93*, annual conference on Computer graphics and interactive techniques, pages 221–230. ACM, 1993.
- [GSG⁺99] Bruce Gooch, Peter-Pike J. Sloan, Amy Gooch, Peter Shirley, and Richard Riesenfeld. Interactive technical illustration. In *I3D '99: Proc. symposium on Interactive 3D graphics*, pages 31–38. ACM, 1999.
- [GSHG92] Gene Greger, Peter Shirley, Philip M. Hubbard, and Donald P. Greenberg. Irradiance Volume. *IEEE Comp. Graph. Appl.*, 18(2):32–43, 1992.
- [GTGB84] Cindy M. Goral, Kenneth E. Torrance, Donald P. Greenberg, and Bennett Battaile. Modeling the interaction of light between diffuse surfaces. In *Proc. SIGGRAPH '84*, annual conference on Computer graphics and interactive techniques, pages 213–222. ACM, 1984.
- [GZ08] Yotam Gingold and Denis Zorin. Shading-based surface editing. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 27(3):1–9, 2008.
- [HBHS05] Vlastimil Havran, Jiri Bittner, Robert Herzog, and Hans-Peter Seidel. Ray maps for global illumination. In *Proc. EUROGRAPHICS Symposium on Rendering 2005*, pages 43–54, 2005.
- [HBR⁺07] Julien Hadim, Tamy Boubekeur, Mickaël Raynaud, Xavier Granier, and Christophe Schlick. On-the-fly appearance quantization on the gpu for 3d broadcasting. In *Web3D '07: Proc. international conference on 3D web technology*, pages 45–51. ACM, 2007.
- [HCA00] Nicolas Holzschuch, François Cuny, and Laurent Alonso. Wavelet Radiosity on Arbitrary Planar Surfaces. In *Proc. EUROGRAPHICS Workshop on Rendering 2000*, pages 161–172. Springer-Verlag GmbH, 2000.
- [HDS09] Jean-Marc Hasenfratz, Cyrille Domez, François Sillion, and George Drettakis. A Practical Analysis of Clustering Strategies for Hierarchical Radiosity. *Comp. Graph. Forum (Proc. of EUROGRAPHICS '99)*, 18(3):221–232, 1999.
- [Hec90] Paul S. Heckbert. Adaptive radiosity textures for bidirectional ray tracing. In *Proc. SIGGRAPH '90*, volume 24 of *annual conference on Computer graphics and interactive techniques*, pages 145–154. ACM, 1990.
- [Hec91] Paul S. Heckbert. *Simulating Global Illumination Using Adaptive Meshing*. PhD thesis, University of California, Berkeley, 1991.
- [Her99] Aaron Hertzmann. Introduction to 3D Non-Photorealistic Rendering: Silhouettes and Outlines. In *ACM SIGGRAPH '99 Course Notes. Course on Non-Photorealistic Rendering*, chapter 7. ACM, 1999.
- [HG04] Martin Hachet and Pascal Guitton. The CAT - When Mice are not Enough. In *Proc. IEEE VR 2004 Workshop: Beyond Glove and Wand Based Interaction*, pages 66–69, 2004. Immersion S.A.S. <http://www.immersion.fr>.
- [Hil54] E. L. Hill. The Theory of Vector Spherical Harmonics. *Am. J. Phys.*, 22(4):211–214, 1954.
- [HKYM00] Hideki Hirayama, Kazufumi Kaneda, Hideo Yamashita, and Yoshimi Monden. An Accurate Illumination Model for Objects Coated with Multilayer Films. In *EUROGRAPHICS 2000 Short Presentations*, pages 145–150. EUROGRAPHICS, 2000.
- [HKYM01] Hideki Hirayama, Kazufumi Kaneda, Hideo Yamashita, and Yoshimi Monden. Visualization of optical phenomena caused by multilayer films based on wave optics. *The Visual Computer*, 17(2):106–120, 2001.
- [Hof98] Donald D. Hoffman. *Visual Intelligence: How We Create What We See*. W. W. Norton & Company, Inc., 1998.
- [Hog91] Burne Hogarth. *Dynallic light and shade*. Watson-Guptill Publications, 1991.
- [HP02] Heinrich Hey and Werner Purgathofer. Real-Time Rendering of Globally Illuminated Soft Glossy Scenes With Directional Light Maps. Technical Report TR-186-2-02-05, Institute of Computer Graphics and Algorithms, Vienna University of Technology, 2002.

- [HPB06] Miloš Hašan, Fabio Pellacini, and Kavita Bala. Direct-to-indirect transfer for cinematic relighting. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 25(3):1089–1097, 2006.
- [HTSG91] Xiao Dong He, Kenneth E. Torrance, François Sillion, and Donald P. Greenberg. A Comprehensive Physical Model for Light Reflection. In *Proc. SIGGRAPH '91*, annual conference on Computer graphics and interactive techniques, pages 175 – 186. ACM, 1991.
- [IA00] Isabelle Icart and Didier Arquès. A Physically-Based BRDF Model for Multilayer Systems with Uncorrelated Rough Boundaries. In *Proc. EUROGRAPHICS Workshop on Rendering 2000*, pages 353–364. EUROGRAPHICS, 2000.
- [IH03] Takeo Igarashi and John F. Hughes. Smooth meshes for sketch-based freeform modeling. In *I3D '03: Proc. symposium on Interactive 3D graphics and games*, pages 139–142. ACM, 2003.
- [IMT99] Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. Teddy: a sketching interface for 3D freeform design. In *Proc. SIGGRAPH '99*, annual conference on Computer graphics and interactive techniques, pages 409–416. ACM/Addison-Wesley Publishing Co., 1999.
- [IP00] Jean-Claude Iehl and Bernard Péroche. An adaptative spectral rendering with a perceptual control. *Comp. Graph. Forum (Proc. EUROGRAPHICS 2000)*, 19(3), 2000.
- [ISO21] Graphic Technology and Photography Colour target and procedures for the colour characterisation of digital still cameras (DSCs), ISO 17321. draft document available at <http://www.cs.sfu.ca/~mark/ftp/IsoWg18/>.
- [JDA07] Tilke Judd, Frédo Durand, and Edward H. Adelson. Apparent ridges for line drawing. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 26(3):19, 2007.
- [JNLM05] Kyuman Jeong, Alex Ni, Seungyong Lee, and Lee Markosian. Detail control in line drawings of 3D meshes. *The Visual Computer*, 21(8-10):698–706, 2005. Special Issue of Pacific Graphics 2005.
- [Joh02] Scott F. Johnston. Lumo: illumination for cel animation. In *NPAP '02: Proc. international symposium on Non-photorealistic animation and rendering*, pages 45–53. ACM, 2002.
- [Kaj86] Jim T. Kajiya. The rendering equation. In *Proc. SIGGRAPH '86*, annual conference on Computer graphics and interactive techniques, pages 143–150, 1986.
- [KAMW05] Anders Wang Kristensen, Tomas Akenine-Möller, and Henrik Wann Jensen. Precomputed local radiance transfer for real-time lighting design. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 24(3):1208–1215, 2005.
- [KBPŽ06] Jaroslav Křivánek, Kadi Bouatouch, Sumanta N. Pattanaik, and Jiří Žára. Making Radiance and Irradiance Caching Practical: Adaptive Caching and Neighbor Clamping. In *Proc. EUROGRAPHICS Symposium on Rendering 2006*. EUROGRAPHICS, 2006.
- [KDMF03] Robert D. Kalnins, Philip L. Davidson, Lee Markosian, and Adam Finkelstein. Coherent Stylized Silhouettes. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 22(3):856–861, 2003.
- [Kel97] Alexander Keller. Instant radiosity. In *Proc. SIGGRAPH '97*, annual conference on Computer graphics and interactive techniques, pages 49–56. ACM/Addison-Wesley Publishing Co., 1997.
- [KGB05] Bertrand Kerautret, Xavier Granier, and Achille Braquelaire. Intuitive Shape Modeling by Shading Design. In *International Symposium on Smart Graphics*, volume 3638 of *Lecture Notes in Computer Science*, pages 163–174. Springer-Verlag GmbH, 2005.
- [KGPB05] Jaroslav Křivánek, Pascal Gautron, Sumanta N. Pattanaik, and Kadi Bouatouch. Radiance caching for efficient global illumination computation. *IEEE Trans. Visualization and Computer Graphics*, 11(5):550–561, 2005.
- [KHR02] Olga Karpenko, John F. Hughes, and Ramesh Raskar. Free-form Sketching with Variational Implicit Surfaces. *Comp. Graph. Forum (Proc. EUROGRAPHICS 2002)*, 21(3):585–594, 2002.
- [KL06] Janne Kontkanen and Samuli Laine. Sampling Precomputed Volumetric Lighting. *J. Graph. Tools*, 11(3):1–16, 2006.
- [KM99] Jan Kautz and Michael D. McCool. Interactive rendering with arbitrary BRDFs using separable approximations. In *ACM SIGGRAPH '99 Conference abstracts and applications*, page 253. ACM, 1999.
- [Kři04] Jaroslav Křivánek. Error Analysis of the Hemispherical SH Projection, December 2004. <http://www.graphics.cornell.edu/~jaroslav/papers/writing/hemisphererror.pdf>.
- [Kv92] Jan J. Koenderink and Andrea J. van Doorn. Surface shape and curvature scales. *Image Vision Comput.*, 10(8):557–565, 1992.
- [Laf96] Eric P. F. Lafortune. *Mathematical Models and Monte Carlo Algorithms for Physically Based Rendering*. PhD thesis, Katholieke Universiteit Leuven, 1996.
- [LBJS07] Julien Lacoste, Tamy Boubekeur, Bruno Jobard, and Christophe Schlick. Appearance preserving octree-textures. In *GRAPHITE '07: Proc. international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*, pages 87–93. ACM, 2007.
- [LF08] Christian Lessig and Eugene Fiume. SOHO: Orthogonal and symmetric Haar wavelets on the sphere. *ACM Trans. Graph.*, 27(1):1–11, 2008.
- [LFD⁺99] Céline Loscos, Marie-Claude Frasson, George Drettakis, Bruce Walter, Xavier Granier, and Pierre Poulin. Interactive virtual relighting and remodeling of real scenes. In *Proc. EUROGRAPHICS Workshop on Rendering '99*, pages 235–246. Springer-Verlag GmbH, 1999.

- [LG07] Florian Levet and Xavier Granier. Improved Skeleton Extraction and Surface Generation for Sketch-based Modeling. In *Proc. Graphics Interface 2007*, pages 27–33. A.K. Peters, 2007.
- [LGS06a] Florian Levet, Xavier Granier, and Christophe Schlick. 3D Sketching with profile curves. In *International Symposium on Smart Graphics*, Lecture Notes on Computer Science. Springer-Verlag GmbH, 2006.
- [LGS06b] Florian Levet, Xavier Granier, and Christophe Schlick. Fast sampling of implicit surfaces by particle systems. In *SMI '06: Proc. Shape Modeling International 2006*, page 39. IEEE Computer Society, 2006.
- [LGS07a] Florian Levet, Xavier Granier, and Christophe Schlick. MarchingParticles: Fast Generation of Particles for the Sampling of Implicit Surfaces. *Computer Graphics & Geometry*, 9(1):18–49, 2007.
- [LGS07b] Florian Levet, Xavier Granier, and Christophe Schlick. Multi-View Sketch-based FreeForm Modeling. In *International Symposium on Smart Graphics*, Lecture Notes on Computer Science. Springer-Verlag GmbH, 2007.
- [LGS07c] Florian Levet, Xavier Granier, and Christophe Schlick. Triangulation of uniform particle systems: its application to the implicit surface texturing. In *WSCG (Winter School of Computer Graphics)*, 2007.
- [LH06] Chang Ha Lee and Xuejun Hao. Geometry-Dependent Lighting. *IEEE Trans. Visualization and Computer Graphics*, 12(2):197–207, 2006. Member-Amitabh Varshney.
- [LLAP05] Grégory Lecot, Bruno Lévy, Laurent Alonso, and Jean-Claude Paul. Master-element vector irradiance for large tessellated models. In *GRAPHITE '05: Proc. international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 315–322. ACM, 2005.
- [LSK⁺07] Samuli Laine, Hannu Saransaari, Janne Kontkanen, Jaakko Lehtinen, and Timo Aila. Incremental Instant Radiosity for Real-Time Indirect Illumination. In *Proc. EUROGRAPHICS Symposium on Rendering 2007*, pages 277–286, 2007.
- [LSSS04] Xinguo Liu, Peter-Pike Sloan, Heung Y. Shum, and John Snyder. All-Frequency Precomputed Radiance Transfer for Glossy Objects. In *Proc. EUROGRAPHICS Symposium on Rendering 2004*, pages 337–344. EUROGRAPHICS, 2004.
- [LSSS06] Paul Lapides, Ehud Sharlin, Mario Costa Sousa, and Lisa Streit. The 3D Tractus: A Three-Dimensional Drawing Board. In *IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TableTop '06)*. IEEE Computer Society, 2006.
- [LW93] Eric P. F. Lafortune and Yves D. Willems. Bi-directional Path Tracing. In *Proc. International Conference on Computational Graphics and Visualization Techniques (Compugraphics '93)*, pages 145–153, 1993.
- [LZK⁺07] Jaakko Lehtinen, Matthias Zwicker, Janne Kontkanen, Emmanuel Turquin, François Sillion, and Timo Aila. Meshless Finite Elements for Hierarchical Global Illumination. Publications in Telecommunications Software and Multimedia TML-B7, Helsinki University of Technology, 2007.
- [LZT⁺08] Jaakko Lehtinen, Matthias Zwicker, Emmanuel Turquin, Janne Kontkanen, Frédo Durand, François Sillion, and Timo Aila. A Meshless Hierarchical Representation for Light Transport. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 27(3), 2008.
- [MAP00a] Sylvain Michelin, Didier Arques, and Benoit Piranda. Modelisation of Implicit Surfaces Driven by an Atlas of Discoids. In *GraphiCon'2000*, pages 256–261, 2000.
- [MAP00b] Sylvain Michelin, Didier Arques, and Benoit Piranda. Overlapping Radiosity: Using a New Function Base with Local Disk Support. In *International Conference in Central Europe on Computer Graphics and Visualization*, volume 2, pages 236–243, 2000.
- [MBPL99] Antoine Manzanera, Thierry M. Bernard, Françoise J. Preteux, and Bernard Longuet. Medial Faces from a Concise 3D Thinning Algorithm. In *ICCV (1)*, pages 337–343, 1999.
- [McC99] Michael D. McCool. Anisotropic diffusion for Monte Carlo noise reduction. *ACM Trans. Graph.*, 18(2):171–194, 1999.
- [MCCH99] Lee Markosian, Jonathan M. Cohen, Thomas Crulli, and John F. Hughes. Skin: A Constructive Approach to Modeling Free-form Shapes. In *Proc. SIGGRAPH '99*, annual conference on Computer graphics and interactive techniques, pages 393–400. ACM, 1999.
- [MFE07] Jason Mitchell, Moby Francke, and Dhabih Eng. Illustrative rendering in Team Fortress 2. In *Symposium on Non-Photorealistic Animation and Rendering*, pages 71–76. ACM, 2007.
- [Mil94] Gavin Miller. Efficient algorithms for local and global accessibility shading. In *Proc. SIGGRAPH '94*, annual conference on Computer graphics and interactive techniques, pages 319–326. ACM, 1994.
- [Mit07] Martin Mittring. Finding next gen: CryEngine 2. In *SIGGRAPH '07: ACM SIGGRAPH 2007 courses*, pages 97–121. ACM, 2007.
- [MKC⁺06] Tom Mertens, Jan Kautz, Jiawen Chen, Philippe Bekaert, and Frédo Durand. Texture Transfer Using Geometry Correlation. In *Proc. EUROGRAPHICS Symposium on Rendering 2006*. EUROGRAPHICS, 2006.
- [MMP02] Rafal Mantiuk, Karol Myszkowski, and Sumanta N. Pattanaik. Cube-Map Data Structure for Interactive Global Illumination Computation in Dynamic Diffuse Environments. In *Proc. International Conference on Compute Vision and Graphics (ICCVG)*, pages 530–538, 2002.
- [MWO03] Wan-Chun Ma, Fu-Che Wu, and Ming Ouhyoung. Skeleton Extraction of 3D Objects with Radial Basis Functions. In *SMI '03: Proc. Shape Modeling International 2003*, page 207. IEEE Computer Society, 2003.

- [Mys97] Karol Myszkowski. Lighting Reconstruction Using Fast and Adaptive Density Estimation Techniques. In *Proc. EUROGRAPHICS Workshop on Rendering '97*, pages 251–262. Springer-Verlag GmbH, 1997.
- [ND05] Marc Nienhaus and Jürgen Döllner. *GPU Gems II*, chapter Blueprint Rendering and Sketchy Drawings, pages 235–252. Addison-Wesley Publishing Co., 2005.
- [NDM05] Addy Ngan, Frédo Durand, and Wojciech Matusik. Experimental Analysis of BRDF Models. In *Proc. EUROGRAPHICS Symposium on Rendering 2005*, pages 117–226, 2005.
- [NDM06] Addy Ngan, Frédo Durand, and Wojciech Matusik. Image-driven Navigation of Analytical BRDF Models. In *Proc. EUROGRAPHICS Symposium on Rendering 2006*, pages 399–408, 2006.
- [NJLM05] Alex Ni, Kyuman Jeong, Seungyong Lee, and Lee Markosian. Multi-scale Line Drawings from 3D Meshes. Technical Report CSE-TR-510-05, Dep. Electrical Engineering and Computer Science, University of Michigan, 2005.
- [NNSK99] László Neumann, Attila Neumann, and László Szirmay-Kalos. Reflectance Models with Fast Importance Sampling. *Comp. Graph. Forum*, 18(4):249–265, 1999.
- [NPG03] Mangesh Nijasure, Sumanta N. Pattanaik, and Vineet Goel. Interactive Global Illumination in Dynamic Environments Using Commodity Graphics Hardware. In *PG '03: Proc. Pacific Conference on Computer Graphics and Applications*, page 450. IEEE Computer Society, 2003.
- [NPG05] Mangesh Nijasure, Sumanta N. Pattanaik, and Vineet Goel. Real-Time Global Illumination on GPUs. *J. Graph. Tools*, 10(2):55–71, 2005.
- [NRH⁺77] F. E. Nicodemus, J. C. Richmond, J. J. Hsia, I. W. Ginsberg, and T. Limperis. *Geometrical Considerations and Nomenclature for Reflectance*. National Bureau of Standards, 1977.
- [NRH03] Ren Ng, Ravi Ramamoorthi, and Pat Hanrahan. All-frequency shadows using non-linear wavelet lighting approximation. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 22(3):376–381, 2003.
- [NRH04] Ren Ng, Ravi Ramamoorthi, and Pat Hanrahan. Triple product wavelet integrals for all-frequency relighting. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 23(3):477–487, 2004.
- [NWT07] Heung-Sun Ng, Tai-Pang Wu, and Chi-Keung Tang. Surface-from-Gradients with Incomplete Data for Single View Modeling. In *ICCV 2007: Proc. IEEE International Conference on Computer Vision*, pages 1–8. IEEE Computer Society, 2007.
- [OBS04] Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel. Ridge-valley lines on meshes via implicit surface fitting. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 23(3):609–612, 2004.
- [OKP⁺08] Juraj Obert, Jaroslav Křivánek, Fabio Pellacini, Daniel Sykora, and Sumanta N. Pattanaik. iCheat: A Representation for Artistic Control of Indirect Cinematic Lighting. *Comp. Graph. Forum*, 27(4):1217–1223, 2008.
- [OMSI07] Makoto Okabe, Yasuyuki Matsushita, Li Shen, and Takeo Igarashi. Illumination Brush: Interactive Design of All-Frequency Lighting. In *PG '07: Proc. Pacific Conference on Computer Graphics and Applications*, pages 171–180. IEEE Computer Society, 2007.
- [ONNI03] Shigeru Ohwada, Frank Nielsen, Kazuo Nakazawa, and Takeo Igarashi. A Sketching Interface for Modeling the Internal Structures of 3D Shapes. In *International Symposium on Smart Graphics*, Lecture Notes in Computer Science, pages 49–57. Springer-Verlag GmbH, 2003.
- [OSSJ09] Luke Olsen, Faramarz F. Samavati, Mario Costa Sousa, and Joaquim A. Jorge. Sketch-based modeling: A survey. *Comp. & Graph.*, 33(1), 2009.
- [PBJ⁺04] João P. Pereira, Vasco A. Branco, Joaquim A. Jorge, Nelson F. Silva, Tiago D. Cardoso, and F. Nunes Ferreira. Cascading Recognizers for Ambiguous Calligraphic Interaction. In *SBIM 2004: EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling*, 2004.
- [PBMF07] Fabio Pellacini, Frank Battaglia, R. Keith Morley, and Adam Finkelstein. Lighting with paint. *ACM Trans. Graph.*, 26(2):9, 2007.
- [PDC⁺03] Timothy J. Purcell, Craig Donner, Mike Cammarano, Henrik Wann Jensen, and Pat Hanrahan. Photon mapping on programmable graphics hardware. In *Proc. SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware*, pages 41–50, 2003.
- [Pee91] Mark S. Peercy. Linear Color Representation for Full Spectral Rendering. In *Proc. SIGGRAPH '91*, annual conference on Computer graphics and interactive techniques, pages 191–198. ACM, 1991.
- [PF95] Pierre Poulin and Alain Fournier. Painting Surface Characteristics. In *Proc. EUROGRAPHICS Workshop on Rendering '95*, pages 160–169. Springer-Verlag GmbH, 1995.
- [PG04] Matt Pharr and Simon Green. *GPU Gems*, chapter Ambient Occlusion, pages 80–83. Addison-Wesley, 2004.
- [PGS07] Romain Pacanowski, Xavier Granier, and Christophe Schlick. Gestion de la complexité géométrique dans le calcul d'éclairage pour la présentation publique de scènes archéologiques complexes. In *Virtual Retrospect 2007 : Archéologie et Réalité Virtuelle*. Ausonius, 2007.
- [PGSP08] Romain Pacanowski, Xavier Granier, Christophe Schlick, and Pierre Poulin. Sketch and Paint-based Interface for Highlight Modeling. In *SBIM 2008: EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling*, pages 11–17. Eurographics, 2008.

- [PGSP09] Romain Pacanowski, Xavier Granier, Christophe Schlick, and Pierre Poulin. Volumetric Vector-Based Representation for Indirect Illumination Caching. Research Report RR-6983, INRIA, 2009.
- [Pho75] Bui Tuong Phong. Illumination for computer generated pictures. *Commun. ACM*, 18(6):311–317, 1975.
- [PL07] Fabio Pellacini and Jason Lawrence. AppWand: Editing Measured Materials using Appearance-Driven Optimization. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 26(3), 2007.
- [PRG⁺08] Romain Pacanowski, Mickaël Raynaud, Xavier Granier, Patrick Reuter, Christophe Schlick, and Pierre Poulin. Efficient Streaming of 3D Scenes with Complex Geometry and Complex Lighting. In *Web3D '08: Proc. international symposium on 3D web technology*, pages 11–17. ACM, 2008.
- [PRL⁺08] Romain Pacanowski, Mickaël Raynaud, Julien Lacoste, Xavier Granier, Patrick Reuter, Christophe Schlick, and Pierre Poulin. Compact Structures for Interactive Global Illumination on Large Cultural Objects. *International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST 2008): Shorts and Projects*, 2008.
- [Pug92] David Pugh. Designing solid objects using interactive sketch interpretation. In *SI3D '92: Proc. symposium on interactive 3D graphics*, pages 117–126. ACM, 1992.
- [PWL⁺07] Minghao Pan, Rui Wang, Xinguo Liu, Qunsheng Peng, and Hujun Bao. Precomputed Radiance Transfer Field for Rendering Interreflections in Dynamic Scenes. *Comp. Graph. Forum (Proc. EUROGRAPHICS 2007)*, 26(3):485–493, 2007.
- [QTG⁺06] Jean-Charles Quillet, Gwenola Thomas, Xavier Granier, Pascal Guitton, and Jean-Eudes Marvie. Using Expressive Rendering for Remote Visualization of Large City Models. In *Web3D '06: Proc. international conference on 3D web technology*, pages 27–35. ACM, 2006.
- [RBD06] Szymon Rusinkiewicz, Michael Burns, and Doug DeCarlo. Exaggerated Shading for Depicting Shape and Detail. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 25(3):1199–1205, 2006.
- [RF91] Maria Raso and Alain Fournier. A Piecewise Polynomial Approach to Shading Using Spectral Distribution. In *Proc. Graphics Interface 91*, pages 40–46. Canadian Human-Computer Communications Society, Morgan Kaufmann Publishers, 1991.
- [RGB⁺03] Holly Rushmeier, Jose Gomes, Laurent Balmelli, Fausto Bernardini, and Gabriel Taubin. Image-Based Object Editing. In *3DIM '03: Proc. Conference on 3-D Digital Imaging and Modeling*, pages 20–28. IEEE Computer Society, 2003.
- [RGKS08] Tobias Ritschel, Thorsten Grosch, Jan Kautz, and Hans-Peter Seidel. Interactive Global Illumination Based on Coherent Surface Shadow Maps. In *Proc. Graphics Interface 2008*, pages 185–192. Canadian Information Processing Society, A.K. Peters, 2008.
- [RH01] Ravi Ramamoorthi and Pat Hanrahan. An efficient representation for irradiance environment maps. In *Proc. SIGGRAPH '01*, annual conference on Computer graphics and interactive techniques, pages 497–500. ACM, 2001.
- [RMB07] Ravi Ramamoorthi, Dhruv Mahajan, and Peter Belhumeur. A First-Order Analysis of Lighting, Shading, and Shadows. *ACM Trans. Graph.*, 26(1):2, 2007.
- [Rus04] Szymon Rusinkiewicz. Estimating Curvatures and Their Derivatives on Triangle Meshes. In *3DPVT '04: Proc. International Symposium 3D Data Processing, Visualization, and Transmission*, pages 486–493. IEEE Computer Society, 2004.
- [SAG94] Brian E. Smits, James R. Arvo, and Donald P. Greenberg. A clustering algorithm for radiosity in complex environments. In *Proc. SIGGRAPH '94*, annual conference on Computer graphics and interactive techniques, pages 435–442. ACM, 1994.
- [SAS92] Brian E. Smits, James R. Arvo, and David H. Salesin. An importance-driven radiosity algorithm. In *Proc. SIGGRAPH 92*, annual conference on Computer graphics and interactive techniques, pages 273–282. ACM, 1992.
- [SAWG91] François Sillion, James R. Arvo, Stephen H. Westin, and Donald P. Greenberg. A Global Illumination Solution for General Reflectance Distributions. In *Proc. SIGGRAPH '91*, annual conference on Computer graphics and interactive techniques, pages 187–196. ACM, 1991.
- [Sbe97] Mateu Sbert. *The Use of Global Random Directions to Compute Radiosity: Global Monte Carlo Techniques*. PhD thesis, Universitat Politècnica de Catalunya, Barcelona, Spain, 1997.
- [SBS99] Sabine Süsstrunk, Robert Buckley, and Steve Swen. Standard RGB Color Spaces. In *The Seventh Color Imaging Conference: Color Science, Systems, and Applications*, pages 127–134. IS&T - The Society for Imaging Science and Technology, 1999.
- [SC04] Amit Shesh and Baoquan Chen. SMARTPAPER—An Interactive and User-friendly Sketching System. *Comp. Graph. Forum (Proc. EUROGRAPHICS 2004)*, 24(3), 2004.
- [Sch94a] Christophe Schlick. An Inexpensive BRDF Model for Physically-Based Rendering. *Comp. Graph. Forum*, 13(3):233–246, 1994.
- [Sch94b] Christophe Schlick. *Graphics Gems V*, chapter A Fast Alternative to Phong's Specular Model, pages 385–384. Morgan Kaufmann Publishers, 1994.

- [SCS⁺08] Larry Seiler, Doug Carmean, Eric Sprangle, Tom Forsyth, Michael Abrash, Pradeep Dubey, Stephen Junkins, Adam Lake, Jeremy Sugerman, Robert Cavin, Roger Espasa, Ed Grochowski, Toni Juan, and Pat Hanrahan. Larrabee: a many-core x86 architecture for visual computing. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 27(3):1–15, 2008.
- [SDS95] François Sillion, George Drettakis, and Cyril Soler. A Clustering Algorithm for Radiance Calculation In General Environments. In *Proc. EUROGRAPHICS Workshop on Rendering '95*, 1995.
- [Sei99] Joshua Seims. Putting the artist in the loop. *SIGGRAPH Comput. Graph.*, 33(1):52–53, 1999.
- [SFCD99] Yinlong Sun, F. David Fracchia, Thomas W. Calvert, and Mark S. Drew. Deriving Spectra from Colors and Rendering Light Interference. *IEEE Comp. Graph. Appl.*, 19(4):61–67, 1999.
- [SFDC00] Yinlong Sun, F. David Fracchia, Mark S. Drew, and Thomas W. Calvert. Rendering Iridescent Colors of Optical Disks. In *Proc. EUROGRAPHICS Workshop on Rendering 2000*, pages 353–364. EUROGRAPHICS, 2000.
- [SGCH94] Peter Schröder, Steven J. Gortler, Michael F. Cohen, and Pat Hanrahan. Wavelet Projections for Radiosity. *Comp. Graph. Forum*, 13(2):141–151, 1994.
- [SGP03] Mark Schroering, Cindy Grimm, and Robert Pless. A New Input Device for 3D Sketching. In *Vision Interface*, pages 311–318, 2003.
- [SH94] Peter Schröder and Pat Hanrahan. Wavelet Methods for Radiance Computations. In *Proc. EUROGRAPHICS Workshop on Rendering '94*. Springer-Verlag GmbH, 1994.
- [SH05] Christian Sigg and Markus Hadwiger. *GPU Gems II*, chapter Fast Third-Order Texture Filtering, pages 313–330. Addison-Wesley, 2005.
- [Sha97] Erin Shaw. Hierarchical Radiosity for Dynamic Environments. *Comp. Graph. Forum*, 16(2):107–118, 1997.
- [She03] Peter Sherley. *Realistic Ray Tracing*. AK Peters, 2003.
- [SHHS03] Peter-Pike Sloan, Jesse Hall, John Hart, and John Snyder. Clustered principal components for precomputed radiance transfer. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 22(3):382–391, 2003.
- [Shi06] Peter Shirley. State of the art in interactive ray tracing. In *ACM SIGGRAPH 2006 Courses*. ACM, 2006.
- [SIP07] Benjamin Segovia, Jean-Claude Iehl, and Bernard Péroche. Metropolis Instant Radiosity. *Comp. Graph. Forum (Proc. EUROGRAPHICS 2007)*, 26(3):425–434, 2007.
- [SKS02] Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 21(3):527–536, 2002.
- [Slo06] Peter-Pike Sloan. Normal Mapping for Precomputed Radiance Transfer. In *I3D '06: Proc. symposium on Interactive 3D graphics and games*, pages 23–26. ACM, 2006.
- [SMGG01] Peter-Pike Sloan, William Martin, Amy Gooch, and Bruce Gooch. The Lit Sphere: A Model for Capturing NPR Shading from Art. In *Proc. Graphics Interface 2001*, pages 143–150. Canadian Information Processing Society, 2001.
- [Smi99] Brian E. Smits. An RGB to Spectrum Conversion for Reflectances. *J. Graph. Tools*, 4(4):11–22, 1999.
- [SP94] François Sillion and Claude Puech. *Radiosity and Global Illumination*. Morgan Kaufmann Publishers, 1994.
- [SP01] Xavier Serpaggi and Bernard Péroche. An Adaptive Method for Indirect Illumination Using Light Vectors. *Comp. Graph. Forum (Proc. EUROGRAPHICS 2001)*, 20(3):278–287, 2001.
- [SRS91] Emanuel Sachs, Andrew Roberts, and David Stoops. 3-Draw: A Tool for Designing 3D Shapes. *IEEE Comp. Graph. Appl.*, 11(6):18–26, 1991.
- [SS95] Peter Schröder and Wim Sweldens. Spherical wavelets: efficiently representing functions on the sphere. In *Proc. SIGGRAPH '95*, annual conference on Computer graphics and interactive techniques, pages 161–172. ACM, 1995.
- [SS98] Marc Stamminger and Hans-Peter Seidel. Three point clustering for radiance computations. In *Proc. EUROGRAPHICS Workshop on Rendering '98*, pages 211–222. Springer-Verlag GmbH, 1998.
- [SSG⁺99] Marc Stamminger, Annette Scheel, Xavier Granier, Frédéric Perez-Carzorla, George Drettakis, and François Sillion. Efficient Glossy Global Illumination with Interactive Viewing. In *Proc. Graphics Interface 99*, pages 50–57. Morgan Kaufmann Publishers, 1999.
- [SSG⁺00] Marc Stamminger, Annette Scheel, Xavier Granier, Frédéric Perez-Carzorla, George Drettakis, and François Sillion. Efficient Glossy Global Illumination with Interactive Viewing. *Comp. Graph. Forum*, 19(1):13–25, 2000. Extended version of [SSG⁺99].
- [SSH⁺98] Philipp Slusallek, Marc Stamminger, Wolfgang Heidrich, Jan Popp-Christian, and Hans-Peter Seidel. Composite Lighting Simulations with Lighting Networks. *IEEE Comput. Graph. Appl.*, 18(2):22–31, 1998.
- [SSS97] Marc Stamminger, Philipp Slusallek, and Hans-Peter Seidel. Bounded Radiosity - Illumination on General Surfaces and Clusters. *Comp. Graph. Forum (Proc. EUROGRAPHICS 1997)*, 16(3):C309–C317, 1997.
- [SSSS98] Marc Stamminger, H. Schirmacher, Philipp Slusallek, and Hans-Peter Seidel. Getting Rid Of Links in Hierarchical Radiosity. *Comp. Graph. Forum (Proc. EUROGRAPHICS '98)*, 17(3), 1998.

- [ST90] Takafumi Saito and Tokiichiro Takahashi. Comprehensible rendering of 3-D shapes. *SIGGRAPH Comput. Graph.*, 24(4):197–206, 1990.
- [Sta99] Jos Stam. Diffraction Shader. In *Proc. SIGGRAPH '99*, annual conference on Computer graphics and interactive techniques, pages 75–84. ACM, 1999.
- [Sta04] Jos Stam. *GPU Gems*, chapter Simulating Diffraction. Addison-Wesley Publishing Co., 2004.
- [Suy02] Frank Suykens - De Laet. *On Robust Monte Carlo Algorithms for Multi-pass Global Illumination*. PhD thesis, Departement of Computer Science, Katholieke Universiteit Leuven, 2002.
- [SW00] Gernot Schaufler and Henrik Wann Jensen. Ray Tracing Point Sampled Geometry. In *Proc. EUROGRAPHICS Workshop on Rendering 2000*, pages 319–328. Springer-Verlag GmbH, 2000.
- [SWSJ05] Ryan Schmidt, Brian M. Wyvill, Mario-Costa Sousa, and Joaquim A. Jorge. ShapeShop: Sketch-Based Solid Modeling with BlobTrees. In *SBIM 2002: EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling*, pages 53–62, 2005.
- [TABI07] Hideki Todo, Ken-Ichi Anjyo, William Baxter, and Takeo Igarashi. Locally controllable stylized shading. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 26(3):17, 2007.
- [TBSR04] Steve Tsang, Ravin Balakrishnan, Karan Singh, and Abhishek Ranjan. A suggestive interface for image guided 3D sketching. In *CHI '04: Proc. SIGCHI conference on Human factors in computing systems*, pages 591–598. ACM, 2004.
- [TBv04] Matthew Thorne, David Burke, and Michiel van de Panne. Motion doodles: an interface for sketching character motion. *ACM Trans. Graph.*, 23(3):424–431, 2004.
- [TDM99] Osama Tolba, Julie Dorsey, and Leonard McMillan. Sketching with projective 2D strokes. In *UIST '99: Proc. symposium on User interface software and technology*, pages 149–157. ACM, 1999.
- [tHR03] Bart M. ter Haar Romeny. *Front-End Vision and Multi-Scale Image Analysis*, volume 27 of *Computational Imaging and Vision*. Kluwer Academic Publishers/Springer-Verlag GmbH, 2003.
- [TL04] Eric Tabellion and Arnauld Lamorlette. An Approximate Global Illumination System for Computer Generated Films. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 23(3):469–476, 2004.
- [TS06] Yu-Ting Tsai and Zen-Chung Shih. All-Frequency Precomputed Radiance Transfer Using Spherical Radial Basis Functions and Clustered Tensor Approximation. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 25(3):967–976, 2006.
- [TZF04] Chiew-Lan Tai, Hongxin Zhang, and Chun-Kin Fong. Prototype Modeling from Sketched Silhouettes based on Convolution Surfaces. *Comp. Graph. Forum*, 23(1):71–83, 2004.
- [van96] Cornelius W. A. M. van Overveld. Painting gradients: free-form surface design using shading patterns. In *Proc. Graphics Interface 96*, pages 151–158. Canadian Information Processing Society, 1996.
- [VBGS08a] Romain Vergne, Pascal Barla, Xavier Granier, and Christophe Schlick. Apparent relief: a shape descriptor for stylized shading. In *NPAR '08: Proc. international symposium on Non-photorealistic animation and rendering*, pages 23–29. ACM, 2008.
- [VBGS08b] Romain Vergne, Pascal Barla, Xavier Granier, and Christophe Schlick. Shading with apparent relief. In *SIGGRAPH '08: ACM SIGGRAPH 2008 talks*. ACM, 2008.
- [VBT⁺07] Romain Vergne, Adrien Bousseau, Joëlle Thollot, David Vanderhaeghe, Pascal Barla, and Xavier Granier. Utilisation du rendu expressif pour l'illustration et l'exploration de données archéologiques. In *Virtual Retro-spect 2007 : Archéologie et Réalité Virtuelle*. Ausonius, 2007.
- [Vea97] Eric Veach. *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, 1997.
- [VG94] Eric Veach and Leonidas J. Guibas. Bidirectional Estimators for Light Transport. In *Proc. EUROGRAPHICS Workshop on Rendering '94*, volume 1, pages 147–162, 1994.
- [VG97] Eric Veach and Leonidas J. Guibas. Metropolis light transport. In *Proc. SIGGRAPH '97*, annual conference on Computer graphics and interactive techniques, pages 65–76. ACM/Addison-Wesley Publishing Co., 1997.
- [VLD07] Peter Vangorp, Jurgen Laurijssen, and Philip Dutré. The influence of shape on the perception of material reflectance. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 26(3):77, 2007.
- [Vol06] Christina Volckman. *Renaissance*, 2006.
- [VPB⁺09] Romain Vergne, Romain Pacanowski, Pascal Barla, Xavier Granier, and Christophe Schlick. Light warping for enhanced surface depiction. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 28(3):25, 2009.
- [VTMS04] Peter A. C. Varley, Y. Takahashi, J. Mitani, and Hiromasa Suzuki. A Two-Stage Approach for Interpreting Line Drawings of Curved Objects. In *SBIM 2004: EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling*, 2004.
- [WAL⁺97] Bruce Walter, Gün Alpay, Eric P. F. Lafortune, Sebastian Fernandez, and Donald P. Greenberg. Fitting virtual lights for non-diffuse walkthroughs. In *Proc. SIGGRAPH '97*, annual conference on Computer graphics and interactive techniques, pages 45–48. ACM/Addison-Wesley Publishing Co., 1997.
- [Wal98] Bruce Walter. *Density Estimation Techniques for Global Illumination*. PhD thesis, Cornell University, 1998.
- [Wan01] Henrik Wann Jensen. *Realistic Image Synthesis using Photon Mapping*. A.K. Peters, 2001.

- [War91] Gregory J. Ward. *Graphics Gems II*, chapter Real Pixels, pages 80–83. Morgan Kaufmann Publishers, 1991.
- [War92] Gregory J. Ward. Measuring and modeling anisotropic reflection. In *ACM SIGGRAPH 92*, pages 265–272. ACM, 1992.
- [WAT92] Stephen H. Westin, James R. Arvo, and Kenneth E. Torrance. Predicting reflectance functions from complex surfaces. *SIGGRAPH Comput. Graph.*, 26(2):255–264, 1992.
- [WCG87] John R. Wallace, Michael F. Cohen, and Donald P. Greenberg. A two-pass solution to the rendering equation: A synthesis of ray tracing and radiosity methods. In *Proc. SIGGRAPH '87*, annual conference on Computer graphics and interactive techniques, pages 311–320. ACM, 1987.
- [WEV02] Gregory J. Ward and Elena Eydelberg-Vileshin. Perfect RGB Rendering Using Spectral Prefiltering and Sharp Color Primary. In *Proc. EUROGRAPHICS Workshop on Rendering 2002*, pages 117–124. Eurographics, EUROGRAPHICS, 2002.
- [WH92] Gregory J. Ward and Paul S. Heckbert. Irradiance Gradients. In *Proc. EUROGRAPHICS Workshop on Rendering '92*, pages 85–98, 1992.
- [WHG99] Andrew J. Willmott, Paul S. Heckbert, and Michael Garland. Face cluster radiosity. In *Proc. EUROGRAPHICS Workshop on Rendering 99*, pages 293–304. Springer-Verlag GmbH, 1999.
- [WHSG97] Bruce Walter, Philip M. Hubbard, Peter Shirley, and Donald P. Greenberg. Global illumination using local linear density estimation. In *Proc. SIGGRAPH '97*, annual conference on Computer graphics and interactive techniques, pages 217–259. ACM, 1997.
- [Wil90] Lance Williams. 3D paint. In *SI3D '90: Proc. symposium on Interactive 3D graphics*, pages 225–233. ACM, 1990.
- [Woo94] Phyllis Wood. *Scientific Illustration*. Wiley, 1994.
- [WRC88] Gregory J. Ward, Francis M. Rubinstein, and Robert D. Clear. A ray tracing solution for diffuse interreflection. In *Proc. SIGGRAPH '88*, annual conference on Computer graphics and interactive techniques, pages 85–92, 1988.
- [WTBS07] Tai-Pang Wu, Chi-Keung Tang, Michael S. Brown, and Heung-Yeung Shum. ShapePalettes: interactive normal transfer via sketching. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 26(3):44, 2007.
- [WTL04] Rui Wang, John Tran, and David Luebke. All-Frequency Relighting of Non-Diffuse Objects using Separable BRDF Approximation. In *Proc. EUROGRAPHICS Symposium on Rendering 2004*, pages 345–354. Eurographics, 2004.
- [WZH07] Rui Wang, Jiajun Zhu, and Greg Humphreys. Precomputed Radiance Transfer for Real-Time Indirect Lighting Using a Spectral Mesh Basis. In *Proc. EUROGRAPHICS Symposium on Rendering 2007*, pages 967–976, 2007.
- [YSv05] Chen Yang, Dana Sharon, and Michiel van de Panne. Sketch-based Modeling of Parameterized Objects. In *SBIM 2005: EUROGRAPHICS Workshop on Sketch-Based Interface*, pages 63–72, 2005.
- [ZCB98] Eric Zeghers, S. Carré, and Kadi Bouatouch. Error-Bound Wavelength Selection for Spectral Rendering. *The Visual Computer*, 13(9-10):424–434, 1998.
- [Zen86] Silvano Di Zenzo. A note on the gradient of a multi-image. *Comput. Vision Graph. Image Process.*, 33(1):116–125, 1986.
- [ZHG08] Hongxin Zhang, Julien Hadim, and Xavier Granier. Using the CAT for 3D Sketching in front of Large Displays. In *International Symposium on Smart Graphics*, Lecture Notes on Computer Science. Springer-Verlag GmbH, 2008.
- [ZHH96] Robert C. Zeleznik, Kenneth P. Herndon, and John F. Hughes. SKETCH: an interface for sketching 3D scenes. In *Proc. SIGGRAPH '96*, annual conference on Computer graphics and interactive techniques, pages 163–170. ACM, 1996.
- [ZIK98] Sergej Zhukov, Andrej Inoes, and Grigorij Kronin. An Ambient Light Illumination Model. In *Proc. EUROGRAPHICS Workshop on Rendering '98*, pages 45–56. Springer-Verlag GmbH, 1998.
- [Zmu92] Michael D. Zmura. Color constancy : surface color from changing illumination. *J. Optical Society of America*, 9(3):490–493, 1992.
- [ZQN95] Ruowei Zhou, Hiok Chai Quek, and Geok See Ng. A novel single-pass thinning algorithm and an effective set of performance criteria. *Pattern Recogn. Lett.*, 16(12):1267–1275, 1995.
- [ŽS03] Roman Ženka and Pavel Slavík. New dimension for sketches. In *SCCG '03: Proc. spring conference on Computer graphics*, pages 157–163, New York, NY, USA, 2003. ACM.
- [ZSP98] Jacques Zaninetti, Xavier Serpaggi, and Bernard Péroche. A Vector Approach for Global Illumination in Ray Tracing. *Comp. Graph. Forum (Proc. EUROGRAPHICS '98)*, 17(3), 1998.

Contents

Introduction	1
1 Sketching 3D Models	5
1.1 Context and Previous Work	5
1.2 Profile Curves for Enhanced Free-Form Modeling	8
1.2.1 Profile-Based Approach	8
1.2.2 Skeleton Extraction	9
1.2.3 Improved Surface Reconstruction	11
1.2.4 Multi-View Interface	12
1.2.5 Discussions	14
1.3 3D Positioning for Large displays	16
1.3.1 Previous Work	16
1.3.2 System Overview	17
1.3.3 Modeling Sessions	18
1.3.4 Results	20
1.4 Conclusion	22
2 Local Illumination and Shading	25
2.1 User-Designed Glossy and Specular BRDF	26
2.1.1 Previous Work	26
2.1.2 User-Controls	28
2.1.3 BRDF Model	29
2.1.4 Results	31
2.1.5 Limitations and Possible Extensions	34
2.2 Shape Depiction through Expressive Shading	36
2.2.1 Previous Work	37
2.2.2 Object-Space Information	39
2.2.3 Image-Space Information	40
2.2.4 Combining Measures	41
2.2.5 Application to Different Shading Styles	42
2.2.6 Simple User-Controls	45
2.2.7 Discussion and Future Work	46

2.3	Conclusion	48
3	Global Illumination	49
3.1	Previous Work	49
3.1.1	Definitions	49
3.1.2	Algorithms	51
3.1.3	Precomputation	54
3.1.4	Energy Representations	55
3.1.5	Independence on 3D Complexity	56
3.1.6	User Edition and Conclusion	57
3.2	Irradiance Vector Grid	58
3.2.1	Directional Irradiance Vector	58
3.2.2	Spatial Reconstruction	59
3.2.3	Precomputation	59
3.2.4	Compression	61
3.3	Applications and Results	62
3.3.1	Geometric Complexity: Precomputation and Hardware Rendering	62
3.3.2	Streaming	64
3.3.3	Caching	67
3.4	Conclusion	69
4	Conclusion & Future Research	71
4.1	Contributions	71
4.2	Future Research: General Approach	72
4.2.1	User-centric Computer Graphics: Definition of Contexts	72
4.2.2	Getting away from Physics	73
4.2.3	Falling back into Physics	74
4.2.4	Personal Involvements	74
4.3	Specific Context: Cultural Heritage	75
A	Global Control on Colored Reflection	77
A.1	Motivations	77
A.2	Previous Work	79
A.3	Bilinear Reflection Model	80
A.4	New Reflection Operators	81
A.4.1	New Color-Space Operators	81
A.4.2	New Color Reflectance Model	82
A.4.3	Possible Simplifications	82
A.5	Coefficients Estimation: $k_{d,e}^c$	82
A.5.1	Coherent Behavior Condition	82
A.5.2	Estimation from Color Matching Functions	83
A.5.3	Estimation for ISO RGB Color Space	83
A.6	Reflection Behavior Control	85
A.6.1	Coefficients Meaning	85
A.6.2	Basis-Dependent Coefficients Adjustment	85
A.6.3	User-Defined Coefficients	85
A.7	Applications and Results	86

A.7.1	Global Illumination	86
A.7.2	Simple Relighting	87
A.7.3	Hardware Accelerated Rendering	88
A.8	Conclusion	89
B	Vectorial Definitions	91
B.1	Definitions	91
B.1.1	Hemispherical Function	91
B.1.2	Directional Radiance Vector	91
B.1.3	Directional Irradiance Vector	92
B.1.4	Better Compression of Directional Vectors	92
B.2	Bases for Directional Vectors	93
B.2.1	Scalar directional basis	93
B.2.2	Vector Basis	94
B.2.3	Comparisons	95
B.3	Extending Radiosity Equation	95
B.3.1	Radiosity Vector	96
B.3.2	Interaction Matrix	96
B.3.3	Vector Radiosity equation	97
B.4	Irradiance Vector for Environment-maps	97
C	Curriculum Vitae	99
	Bibliography	109