Modeling View-Dependent Surface Point Radiance with Parameter Maps

Bjarne Kondrup Mortensen & Jens Rosenkjær Andersen

Faculty of Engineering, Science and Medicine
Aalborg Universitet

Laboratory of Computer Vision and Media Technology

TITLE:

Modeling View-Dependent Surface Point Radiance with Parameter Maps

PROJECT PERIOD:
Computer Vision and Graphics,
10th Semester

February 1st 2007
- June 7th 2007

PROJECT GROUP:
CVG 1020

GROUP MEMBERS:
Bjarne Kondrup Mortensen
Jens Rosenkjær Andersen

SUPERVISOR:
Claus Brøndgaard Madsen

PUBLICATIONS: 4

RAPPORT PAGES: 109

APPENDIX PAGES: 19

TOTAL PAGES: 145

ABSTRACT:

In this thesis it is investigated how real world surfaces can be sampled, modeled and reconstructed in computer graphics applications where properties like view-dependent appearance and subtle geometric details are preserved.
A dense grid of sampling points on the surface are photographed from multiple view directions using high dynamic range imaging. The recorded radiances from each surface point are saved.
Models are fitted to the obtained data in order to achieve a unique and compact parametric representation of the radiance distribution on a per-point basis. Two models are employed: a modified Phong reflection model and spherical harmonics. The feasibility of these models are investigated in tests.
Custom vertex and fragment shader programs are implemented in OpenGL to be able to reconstruct the complex surface using simple geometry. The parameters and shader programs to the corresponding model are loaded onto the graphics card and used to recreate the view-dependent appearance of the surface in an interactive real time application.
Tests show strengths and weaknesses of both models and experience is gained regarding how the sampling should be performed for optimal results.

Preface

This master's thesis is carried out at the 10$^{\text{th}}$ semester of Computer Vision and Graphics, during the spring of 2007. The main theme is "*Computer Vision and Computer Graphics*", where the subject of "Modeling View-Dependent Surface Point Radiance with Parameter Maps" is chosen.

The purpose of the work is to apply knowledge obtained through courses during earlier semesters in a problem oriented and project organized form.

Structure and Contents

The thesis starts with an introduction and a preview of the developed system. The remainder of the thesis is structured in six parts, where each part treats part of the process of answering the problem statement. The parts are:

I. a preliminary analysis investigating related work and inventive texturing methods, before presenting an overall design strategy and the problem statement including the goals for the work.

II. a part regarding how the radiance from surfaces are sampled and how the acquired data is treated.

III. a part that details how models are fitted to the acquired data. Two models are investigated and comparative tests are performed.

IV. a part where the process of exploiting the programmability of modern graphics hardware for visualizing the radiance of the surface is described.

V. an evaluative part that tests the system as a whole before concluding upon the work and suggesting directions for further work.

VI. a number of appendices supporting the thesis by detailing relevant subjects. References to an appendix is made whenever more information regarding a topic can be found there.

Reading Guidelines

References to content on the DVD are given as e.g. (◎`/Test/`), for the folder “Test” in the root of the DVD. The content of the DVD is described in Appendix G.

Equations and formulas use the notation **a** for vectors and **A** for matrices.

References to sources are marked with author name and year, e.g. [Kaj86].

Authors

Bjarne Kondrup Mortensen

Jens Rosenkjær Andersen

Contents

1 Introduction

It is a well-known fact that the appearance of surfaces of real world objects changes in relation to the direction from which they are viewed. The way a surface looks in some context is roughly a combination of the light that is present, where the light is placed, the properties of the surface and view direction. An example of this is given in Figure 1.1, where a book lying on a table in front of a window is shown from three different angles. The pictures are taken in immediate succession and show that the appearance of a surface can change much, only as a result of changing view direction. Furthermore, very discernable highlights appear when viewing the surface in the direction that light is reflected.

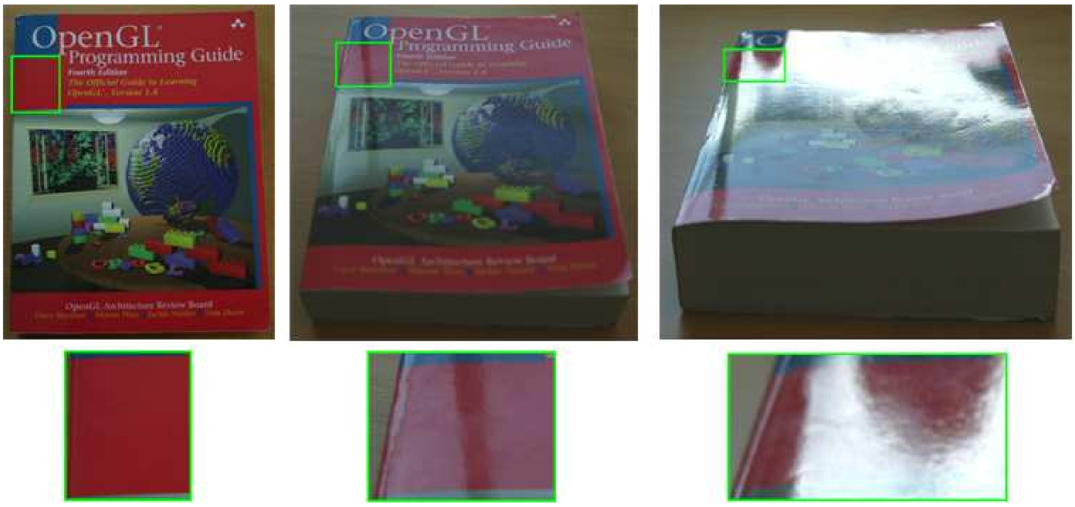


Figure 1.1: Three images of a book lying on a table in front of a window taken in immediate succession. The difference in surface appearance is clear. An example patch of the surface is marked with green boxes and enlarged beneath each image for comparison.

Normally in computer graphics applications, such as games, real world surfaces are recreated using textures. A texture is normally an image which can be pasted onto a surface of an object. Should the surface from the book in the particular scene above be reproduced in a computer graphics application, it is clear that it is not enough to just take one picture of the book and use this to represent its appearance seen from all directions. If the image is taken from an oblique view with highlights, the cover of the book is hard to read. If the image is taken from above so the cover can be read, no information regarding the highlighting is present.

The current prevalent computer graphics solution to this inadequacy is to paste the image onto the surface, and let this represent its base color appearance. Then light sources are explicitly modeled, the surface is assigned overall reflection properties and a shading model is determined upon. However, it can not be guaranteed that the reflection properties are uniform over the entire surface. It is furthermore quite cumbersome and difficult, if not impossible, to model everything realistically. It is e.g. impossible to photograph the diffuse albedo of a specular surface.

Hypothetical Task: Creating a Real Time Virtual Walkthrough of a Real Place

The hypothetical task of creating an interactive, real time walkthrough of an existing real place is considered in the following. This could be a building of a certain significance. This is done to present two overall approaches to the problem of recreating a real place in computer graphics.

Modeling Everything Explicitly One approach is to create 3D models of everything in the given environment: walls, floors, doors, windows, furniture etc. This is a very cumbersome task and would require enormous amounts of time, but is not impossible.

For the 3D models to look realistic, it is also necessary to model all surface materials: wood, glass, marble, plastic, metal and so on. This is an even more difficult and cumbersome task, because the objects might consist of many different materials in conjunction, which each require specific and explicit modeling.

Furthermore, it is necessary to model all the light sources in the scene; their form, appearance, light emission properties etc. Light sources can be quite complex in their nature, e.g. when designed to have some aesthetically pleasing form.

When having all of the above modeled correctly, the scene needs to be rendered using some global illumination solution. Global illumination is a task not yet solved for real time applications, which presents the choice between having slow interaction or settling with a real time rendering technique. None of the choices are really applicable, if the given task is to be fulfilled completely.

Taking Pictures of Everything A radically different approach is to simply take pictures of everything and present the user with a picture that is taken from where he is located. This would not require any explicit modeling of neither the objects, their materials nor the light influencing the scene.

However, there is one simple and very apparent problem with this. To take enough pictures of everything from all positions and in all directions requires a ridiculously large number of pictures. It borders what is practically possible, both because of storage requirements and the time it would take to record all the pictures.

One solution would be to take pictures of the interesting sites of the scene. This is strictly speaking not a viable solution, because it would limit the user in his movement in the virtual recreation of the real place. Only the positions and directions that were used to capture the images would be available to the user. Interpolation would relax this constraint, and allow the user to move more freely, but only within the limits imposed when recording images of the scene. This approach is called Image Based Rendering (IBR), and is used in various commercial applications, but it is still not a complete solution to the hypothetical task set up in this example.

The Best of Both Worlds Two approaches to achieving the same goal are presented in the previous sections. They seem radically different, but could draw use from some of the ideas present in the opposing method. The methods are presented as opposites, and there seems to be a need for some method which can combine the best of both worlds.

It would be advantageous to be able to combine the 3D modeling of the objects with some image based model of the global illumination present in the scene. It might also be possible to employ inventive methods for representing subtle details in the models of the objects without actually modeling things like small cracks or wrinkles in some surface materials. By both modeling the scene as well as recording images of it and investigating how these techniques can be used in conjunction, the first stepping stones towards enabling real time, user-controlled walkthroughs of virtually recreated scenes might be laid.

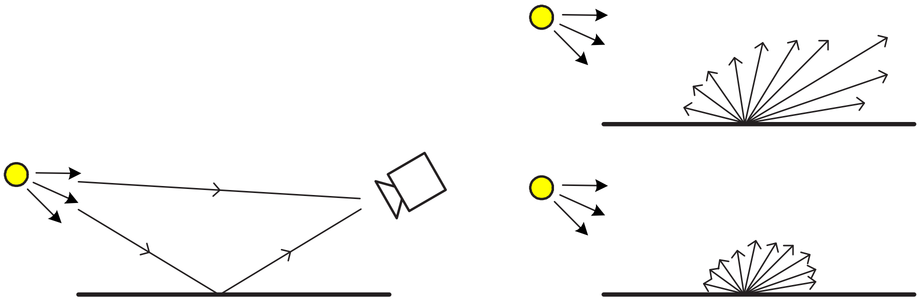
Sampling Real World Surfaces

To be able to recreate a real world surface in a computer graphics application, no matter which of the above approaches is chosen, it is necessary to record it and investigate how it looks and behaves. This is normally done using cameras, taking pictures of the real world. A digital camera essentially records light, or radiance, at a number of pixels on an internal chip. These recorded radiance intensities can be reproduced as an image on e.g. a computer screen. Figure 1.2(a) shows examples of paths that rays from a light source can travel to a camera. The light entering the camera directly from the light source will have the same color as the light source itself. The color of the light that is reflected off a surface before entering the camera is a combination of the light and surface colors [1]. The way it is combined is defined by the reflection properties of the surface.

Reflection properties of surfaces can vary much. One extreme is predominantly diffuse surfaces such as matte paper or dry, unpolished stones where reflection of light depends on the position of the light and not the view direction. In the other end there are highly specular surfaces such as mirrors, where the perceived intensity depends very much on the position of the viewer. Figure 1.2(b) shows how a point on a surface radiates light differently, dependending on the material. The upper figure exemplifies a glossy surface, with higher radiance in the mirror direction. The lower figure shows a more diffuse surface, with an almost isotropic outgoing radiance.

In computer graphics applications, surfaces can typically be assigned a material, where a number of parameters determine how it reacts to incoming light. These parameters typi-

[1] See Appendix B for more elaboration on colors and recording thereof.



(a) The light that hits a camera can come directly from a light source or from the reflection of light on a surface.

(b) The intensity of light leaving a point depends on light sources and their relative position to the point, but also on the properties for the surface.

Figure 1.2: Figures showing paths that light can travel from a light source to a camera, as well as how light can reflect differently from different materials of surfaces.

cally seek to replicate the properties of some real world material. To light a scene correctly, it is necessary to know how all lights and surfaces interact with each other with respect to absorbing, emitting and reflecting light. This is expressed in the "rendering equation"[2]. This equation relies on light and surface parameters to calculate mutual reflection between light and surfaces but also surfaces in between. Modeling this interaction explicitly and realistically is a very hard task not suited for real time applications.

Instead of trying to model this interaction, a contrary approach is to let Mother Nature do her part of the work and then record a chosen surface by extensively taking pictures of it - i.e. sample its appearance from different view directions under some given static illumination conditions. Images capture the result of the natural global illumination solution and do therefore not rely on any parameters. When having sampled the surface from a high number of directions, the obtained information can then be used in different ways to reproduce the appearance of the surface in a computer graphics application.

The Optimal Result and Its Applications

The optimal goal is the ability to recreate arbitrary real world surfaces completely and correctly. This is recreation of a global illumination solution without explicitly modeling light sources and without being bound to some shading model. The optimal solution would include not only diffuse surfaces, but also mirror-effects and the ability of reproducing subtle surface details which are only visible from view directions.

Given that this goal is achieved, a method would be provided for computer graphics applications to produce surfaces whose radiance appear view-dependent in a true-to-life manner. There are two apparent uses for such surfaces. One is to generally employ more realistic

[2] See Appendix E for more information on the rendering equation.

surfaces that mimic real world properties. The other is employing view-dependent surfaces as more complex light sources than the usual directional-, spot- and point light sources available in e.g. OpenGL.

Initiating Problem

Based on the above, this thesis takes its starting point in seeking to investigate how real world surfaces reflects and radiates light and how this can be recreated in virtual 3D contexts. The initiating problem for this thesis is thus expressed in the following question:

Is it possible to capture view-dependent radiance information of a surface and recreate it in a real time computer graphics application?

System Preview 2

The purpose of this chapter is to present a short, informative preview of the work and methods in this thesis. The stages involved in the work that are documented in the report are presented, and a look ahead into the achieved results is given.

The introduction in the previous chapter explained the problem area of reconstructing real world surfaces realistically. Based on the initiating problem, it is set out to investigate the possibilities of capturing real world, view-dependent surface radiance information and recreating it in computer graphics applications. This would be a step towards being able to model true-to-life global illumination solutions of scenes without explicitly modeling things such as light sources or subtle details in geometry.

Methods

The methods employed first involves sampling the surface that it is desired to recreate. This is done by photographing it from hundreds of view directions. Two images of a real world surface are show in Figure 2.1. The images are taken with the same exposure time,

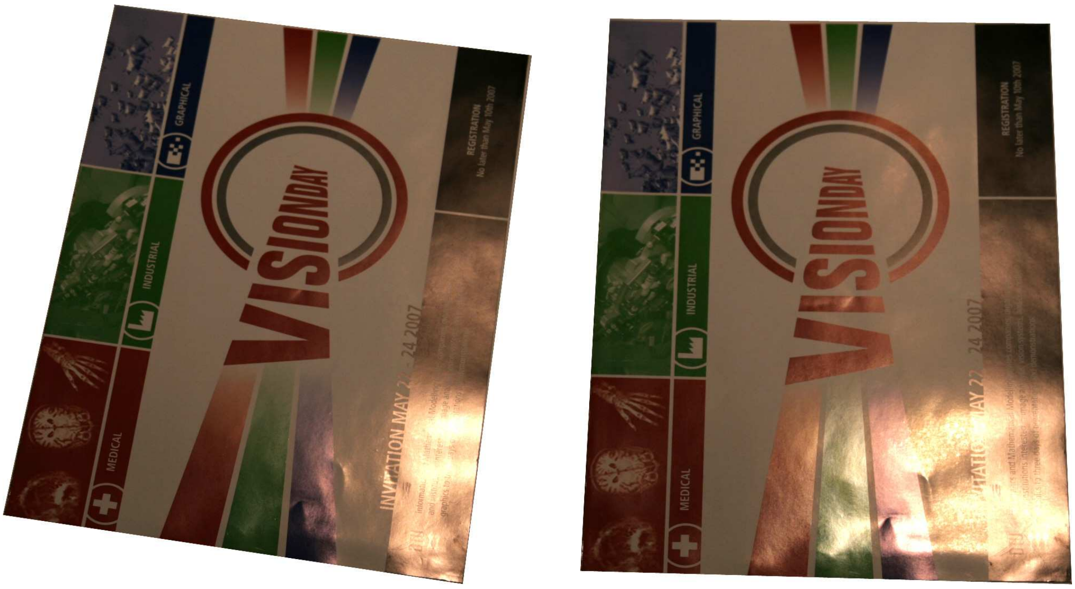


Figure 2.1: Two photographs of the same surface seen from slightly different views with an equal exposure time of 3.2 seconds. The differences are noticable in subtle material details in the highlights.

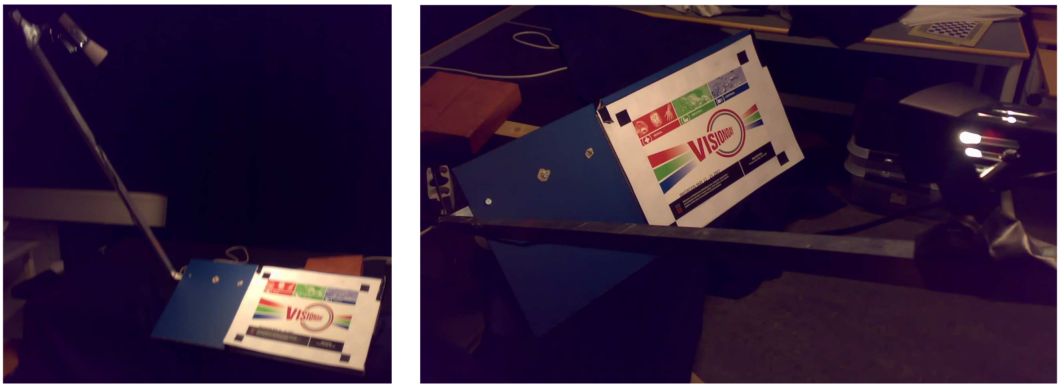


Figure 2.2: The setup used to acquire samples of the surface from a lot of directions.

from slightly different angles. The view-dependent changes in the appearance of the surface are noticeable in the specular parts in the lower right part of the images. The sampling can be performed on both synthetic and real-world data. Real world data is recorded using high dynamic range imaging.

A rig has been build to ease the sampling procedure, where the surface is fixed on a rig where it can be rotated around two axes. It is also possible to mount a light on the right, to keep it in a static position relative to the surface. Two pictures are shown of the rig in Figure 2.2.

To be able to recreate subtle details of the surface, the surface is divided into a dense grid of sample points, and the radiance from each of these points is recorded from each of the sampled view directions. When having obtained all observations of all surface sample points from all view directions, two different models are investigated. This is done to be able to parameterize the view-dependent radiance of the surface on a per-point basis and obtain a compact representation of it. It requires significantly less space to save the model parameters than saving the actual measured surface radiances.

By parameterizing the surface point radiance distribution, it is viable to implement custom shader programs that can recreate the view-dependency of the overall appearance of the surface. Custom vertex and fragment shader programs are implemented in OpenGL for each of the employed radiance distribution models. An interactive application is developed where the user can explore the view-dependency of the reconstructed surface.

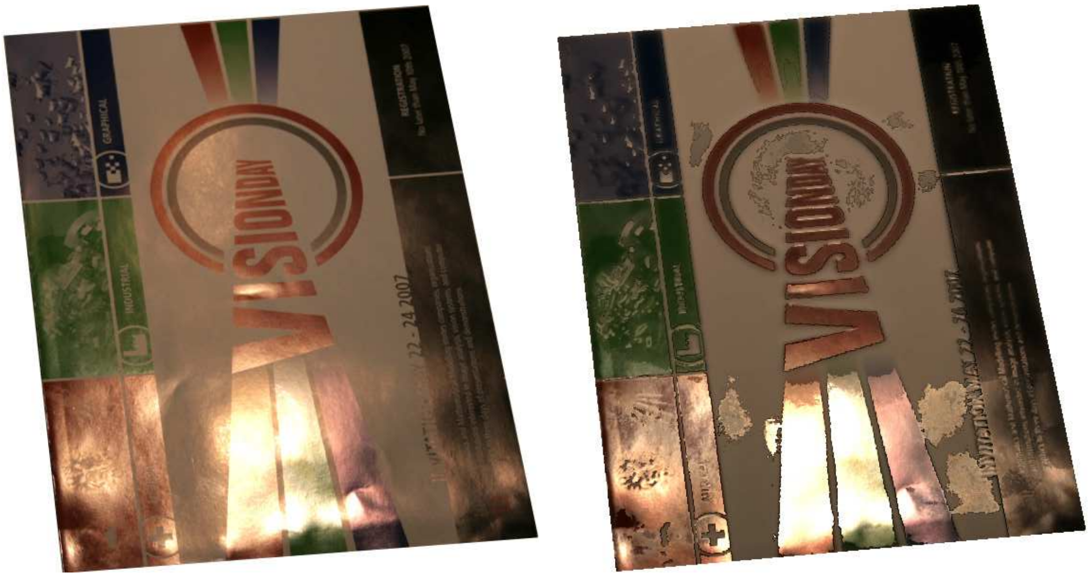
Achieved Results

Capturing, modeling and reconstruction of real world surfaces is accomplished in this thesis. It is possible to sample surface densely enough and from enough view-directions, so that subtle details such as wrinkles or cracks can be reconstructed. Both these phenomena are highly dependent on the direction from which the surface is viewed. No explicit modeling of any light sources or complex geometry is used to recreate the surfaces in a true-to-life manner.

Figure 2.3 show a real world image on top of a reconstruction of the surface from the same point of view. The surface is a paper flyer for a conference. The paper is a glossy, coated type which reflects light quite much. Furthermore, subtle bends and wrinkles are present, giving the surface some very non-uniform reflection appearance.

It is evident that the result shown here does not indicate perfect reconstruction of the real surface. Investigating discrepancies between sampled and reconstructed data is one of the main topics of tests in this thesis.

An executable application showing all the investigated models and surface can be found on the DVD (◎`/Application/executable/`), the reader is urged to explore this interactive application as it helps gain insight to the work and results thereof. A reference on how to use the application is given in Appendix G.



(a) Photograph of surface.

(b) Reconstructed surface, Phong model.

Figure 2.3: A photograph of a surface with different types of material on top of a reconstructed version of the same surface using the modified Phong model.

Part I
Preliminary Analysis

The preliminary analysis takes its starting point in the initial problem for this work and has the aim of formulating a problem statement and enumerating goals for the work.

The first part of the analysis presents a survey of related work within the field of recording, representing and recreating certain properties of surfaces in computer graphics. The result of the survey is a comparison of two overall approaches.

The next chapter reviews different texturing methods. This is included because the method of applying a parametric model at fragment level resembles some of the more sophisticated texturing methods. A review of texturing methods show what is currently possible with imaginative use of textures in computer graphics applications.

The part continues with a design chapter that coarsely lays the stepping stones for the rest of the work and defines interfaces between the different stages.

Following this, the problem statement sums up the main findings from the analysis, before stating the goals for the work. An outline of the rest of the thesis is also presented.

Related Work 3

In this chapter, four examples of related work within the field of recording and recreating surface properties are listed and described. Lastly, a comparison of the two overall approaches is presented. The related work described in this chapter is that which was found to be most prevalent. This survey is carried out to form a basis for the work behind this thesis, as well as to gain inspiration.

3.1 Image Based Rendering

The scene from "The Matrix" where Neo dodges bullets from one of the agents in slow motion and the camera rotates around him is a good example of Image Based Rendering (IBR). Multiple cameras shoot the same scene simultaneously and the end result gives the impression that time is slowed down and the camera rotates seamlessly around Neo. The same technique was used during Super Bowl XXXV in 2001, to provide a unique 3D view of selected plays, which could then be paused to seamlessly rotate the camera to a new angle before continuing.

The survey by [ZC04] presents IBR in relation to the "plenoptic function" [AB91]. The plenoptic function is a 7D function that models an environment in 3D, seen from all positions $[V_x, V_y, V_z]$, in all directions $[\theta, \phi]$, at all times $[t]$, over any range of wavelengths $[\lambda]$. This is an extremely general representation of an environment, and corresponds to a full "appearance description" of the environment which is the opposite of a "source description" comprised of object models, textures, surface reflection properties etc.

IBR is a two stage concept: sampling and rendering. First, a number of samples from the plenoptic function, usually in the form of images, are acquired. Using these samples, it is sought to reconstruct the plenoptic function to be able to seamlessly move between e.g. positions, view directions or points in time. It is far too extensive to sample the entire plenoptic function, which is why one or more limitations are usually imposed upon it. An example is static scenes, for which the camera can be freely moved and the time-parameter can be dropped. Furthermore, the viewer can be restricted to moving along a path, and more dimensions can be dropped. These restrictions seem reasonable for "virtual touring", where the viewer follows some pre-defined path when exploring a large scene.

3.2 Surface Light Field

One 4D version of the plenoptic function is a Surface Light Field (SLF). An SLF discards λ, t and V_z, due to sampling on a surface in a static scene with an RGB camera. It is a function that "assigns an RGB-value to all rays leaving all points on a surface" [WAA+00].

SLF is a variation of the light field [LH96]. The light field has its roots in environment maps, which are recordings of incident light arriving from all directions at a point. The original proposal to representing the 4D light field is by use of a "light slab" where two convex quadrilaterals are used to parameterize the rays of light entering one square and exiting another. This method requires no knowledge about the geometry of the scene. A similar approach is the "Lumigraph" [GGSC96], where coarse knowledge about the scene geometry is employed. The two methods were presented in the same proceedings.

The SLF is thus an approximation to the plenoptic function sampled on an image plane where interpolation is used for reconstruction of novel views. The goal of [MRP98] is to allow unconstrained walkthroughs of static scenes shaded with results of precomputed global illumination. The method captures light leaving a surface at point $[U, V]$ in the direction $[S, T]$, in relation to the light field, where $[U, V]$ and $[S, T]$ represent points on the two quadrilaterals of the light slab.

The SLF handles surface texture, rapid variation in specular highlights and global effects such as interreflection and shadows. Research has been done with the aims of improving the SLF in various ways. An example is [WAA+00], where the SLF is extended to become editable. This allows for deformation of the objects, relative movements and change in the properties of the surfaces at run time.

3.3 View-Dependent Texture-Mapping

The process of texturing polygons of a scene to construct a novel view is known as View-Dependent Texture-Mapping (VDTM). It is a method presented by Debevec, Taylor and Malik and employed in [DBY98]. It is an IBR technique which makes use of projective texture mapping, where reasonably accurate geometric knowledge of the scene is needed.

VDTM works by taking pictures of a scene with known geometry and seeks to construct novel views of the scene by interpolation of up to 3 recorded views. It is a simplification of the light field approach, with much sparser sampling but more complex geometry.

The algorithm involves precomputation of a "view map" for all polygons visible in more than 1 view. This map contains information about which polygons are visible from what views, and is used later on for interpolation. For polygons that are not visible in any views, an object-space hole filling algorithm is employed to avoid flickering between frames.

Lastly, when viewing the scene, three render passes are used to reconstruct the novel view of a polygon. This comprises a mix between the three views closest to the currently desired view position.

3.4 Bidirectional Texture Function

Early attempts to capture the appearance of real material surfaces was done by [DvGNK97]. The work compiled a large, free database with over 12,000 images constituting 61 surface materials with both isotropic and anisotropic appearance. The data is stored as Bidirectional Texture Functions (BTFs), which are functions describing the appearance of a surface at fine-scale level with varied viewing and illumination directions.

To exemplify and prove functional utilization of the data in [DvGNK97], it is fitted to the low-parameter Oren-Nayar Bidirectional Reflectance Distribution Function (BRDF), [ON95], and the Koenderink et al. BRDF representation, [KvDS96], with different orders, resulting in use of 5 and 55 parameters. In contrast to standard color or albedo texture mapping methods, this method can represent the complex dependencies on viewing and lighting directions of rough surfaces. As stated in [SvBLD03], the BTF can handle subtle lighting effects, such as self-shadowing and -reflection.

3.5 Summary

To coarsely summarize, the preceding survey of related work within this field suggests two approaches: either to sample a surface with the aim of fitting a parametric model to the appearance of the surface, or to interpolate between available sampled views for construction of a novel view. Both have their pros and cons.

The view-dependent appearance of a surface depends on the placement and properties of light sources present in the scene as well as properties for the surface in question. The two approaches have different views on how this should be represented. By interpolating between recorded views, no explicit representation of light sources and material properties is saved, but is implicitly encoded in the images. In the other approach, where a model is fitted to the data for each point of the surface, light source and material information is encoded in the model, e.g. as a higher radiance in a reflection direction. This constitutes a more explicit light and material representation than the interpolation approach, but still without explicitly modeling any light sources, material properties or surface shading.

An immediate disadvantage of interpolating between all the taken images of a scene is that all the images would have to be saved, resulting in a need for storing large amounts of data. If the data is fitted to a parametric model, only the parameters need to be saved to represent the data. By saving only the parameters of a model, a relatively compact representation of the data is obtained. Moreover, if a surface exhibits uniform radiance properties across its entire area, it might be suitable to represent this using only one overall model, which is very compact in contrast to saving tens or hundreds of images of it.

Fitting a model to some data inevitably imposes some degree of data loss. The fewer parameters used to represent the data, the larger the risk of losing important details. This would not happen if all the images were saved. On the other hand, the data can also be faulty in various ways or contain noise, and it is undesirable to use bad data in interpolation. If a model is fitted to the data, it might be possible for it to filter out erroneous or noisy samples.

When fitting a model to data, there is also always the question of how well-suited the model is to the data. If an inappropriate model is chosen, it might be error prone to force it to fit the data. If all data is saved, i.e. all images, nothing is forced to fit anything. Thus, it is necessary to determine whether or not a compact representation or a complete set of data is most important.

Another way of comparing the two methods is to look at how fast they are able to reconstruct the real world surfaces. Without optimization, the VDTM method presented in [DBY98] could run with 20 frames per second, requiring three render passes. If a model is fitted to the sampled points on the surface, it is well-suited for implementation on the Graphics Processing Unit (GPU). The parameters for the model can be packaged into textures and a custom shader program can read these values and recreate the correct radiance to each of the points. This requires only one pass and is likely to go very fast, but naturally depends on the number of surface points for which a model needs to be reconstructed and the complexity of the model.

Texturing Methods 4

Texturing has become more than just the task of pasting an image onto a surface. This chapter reviews and describes some of the most prevalent texturing methods in current computer graphics to investigate the possibilities of inventive texturing.

As mentioned in Section 3.5, an advantage for the model fitting approach is that it is compact for uniform surfaces. Such surfaces exhibit properties that change very little over their area, and can thus be represented with one model for the surface.

The more frequent the appearance of the surface, the more it properties changes over its area. One way to represent this is to mesh the surface and represent each mesh polygon or vertex with its own model. At the finest scale, a parametric description is present for each fragment of the surface being rendered. This can be represented with textures. In its classic form, texturing is the process of pasting an image onto a surface of a 3D model. However, more inventive texturing methods are available due to the advent of programmable shaders. The following sections briefly investigates some of the most prevalent texturing methods.

4.1 Image Texture Mapping

As a step towards reproducing the real world in computer graphics applications, the concept of texture mapping was invented. The work was pioneered by [Cat75], and is now very commonly used. The process allows a picture of e.g. a real world object to be pasted onto geometry in a 3D application. An example is given in Figure 4.1, where a picture of some bricks is pasted onto two polygons.

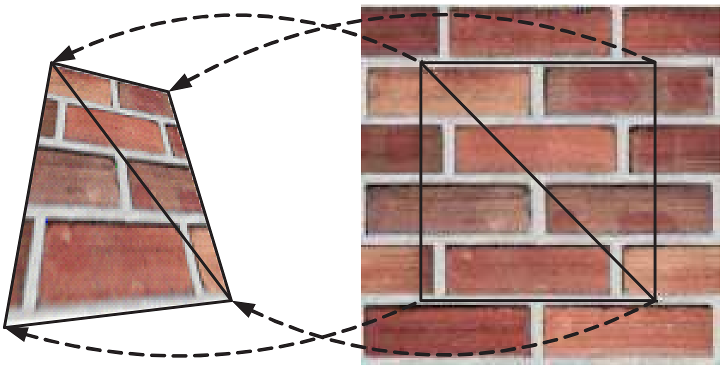


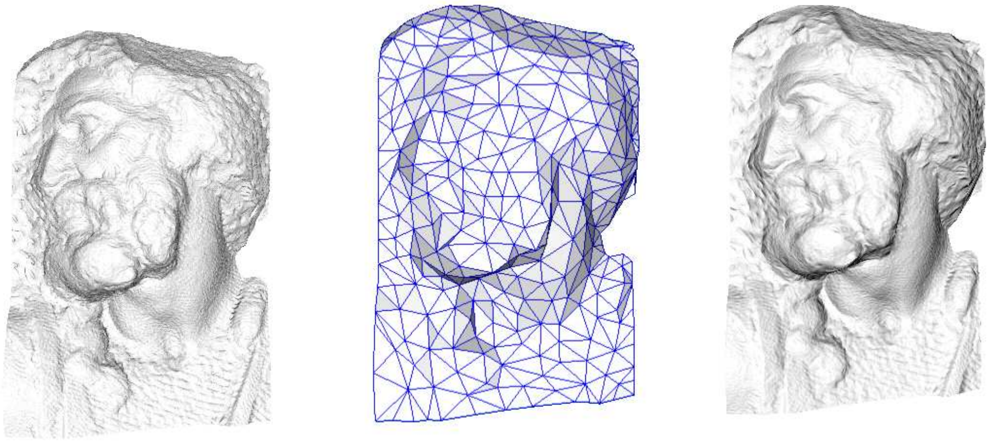
Figure 4.1: A texture (right) is mapped onto two polygons (left).

Texture mapping presented a leap forward towards more realistic computer graphics. Instead of coloring polygons using a single color, or interpolation of per-vertex defined colors, it was now possible to show detailed reproductions of the real world. Of course, texture mapping a real world image of a non-diffuse surface onto an object is limited to correctness for one single view direction. As soon as the view changes, the real-world correspondence is disturbed. As seen in the following sections, restrictions towards what can be achieved with texture mapping is alleviated more and more as research within the field carries on.

4.2 Normal Mapping

Normal mapping is an extension to bump mapping, a method presented in [Bli78]. In short, bump mapping is a perturbation of surface normals to represent minor unevenness of surfaces. The bump map is essentially an image indicating the extend of the perturbation. The perturbation changes how the object reacts to lighting, but it does not change the geometry of the object.

Normal mapping is a common bump mapping technique. In contrast to conventional bump mapping, the entire normal vector is replaced, and its components are saved in an RGB-image. This way, the lighting calculation for each fragment uses the normal in the map instead of the geometry normal, resulting in more detailed shading.



(a) Original mesh, 4.000.000 polygons.

(b) Simplified mesh, 500 polygons.

(c) Simplified mesh, 500 polygons - with normal mapping.

Figure 4.2: Figures illustrating the concept of enhancing low-polygon models using normal mapping. Figures from `http://en.wikipedia.org/wiki/Normal_mapping`.

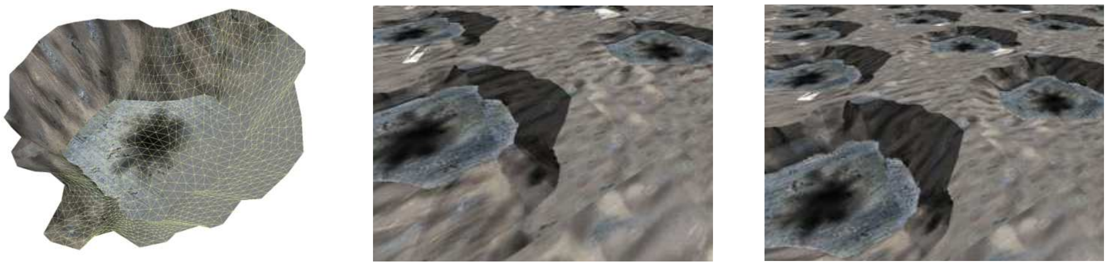
Like bump mapping, normal mapping seeks to add detail to shading without changing the geometry itself. [CMSR98] and [COM98] both present methods for using normal mapping to enhance a low-polygon model by use of normals calculated on the basis of a high-polygon model of the same object. An example of how well a low-polygon model can be enhanced is shown in Figure 4.2.

A radically different approach is displacement mapping. Displacement mapping involves a perturbation of the geometry of an object, instead of the normal. Here, a height map is also employed. It is a method that allows for great sense of depth and details. However, it is a costly procedure because of the likely need for modification and addition of geometry to allow the mesh to be displaced accordingly and on the fly.

4.3 Parallax Mapping

Motion parallax is the relative change in position of two stationary points as seen by a moving observer. Thus, parallax mapping, proposed in [KTI$^+$01], is a method to alleviate the restriction of normal textures to one correct viewing direction. Parallax mapping is also commonly used to give the impression of depth using a 2D texture on some surface.

A depth and a normal map is stored along with the texture map. The depth map can e.g. come from a detailed 3D model of the phenomena which the parallax map is supposed to simulate. Figure 4.3(a) shows such a model for a bullet hole, which is to be put on a surface as a decal to simulate the user having shot it. The combination of a normal map with a depth map in the alpha channel is also called a "relief map".



(a) A detailed 3D model used for construction of the parallax map.

(b) The effect of bump mapping.

(c) The effect of parallax mapping.

Figure 4.3: Figures illustrating the concept and effect of parallax mapping. Figures from `http://cowboyprogramming.com/?p=34`.

Figure 4.3(b) shows a surface with the image of a bullet hole pasted on using standard bump mapping techniques. Figure 4.3(c) shows the end result of rendering "fake" holes in a surface using parallax mapping. Note that for the optimal experience, one should move around the parallax map to perceive the change in its appearance.

To obtain the parallax map effect for a pixel, a few simple steps are needed: first the position of the pixel in the texture, $[u, v]$, is found, next the depth at that position is read and used to scale a vector from the world position of the pixel to the viewer, the $[x, y]$-components of the scaled vector are added to the $[u, v]$ coordinates to obtain a new position, $[u', v']$, in the texture map which is then used as the final color of the pixel.

4.4 Summary

The above sections review a number of different methods for texturing surfaces to make them appear more realistic than just having some uniform or vertex-interpolated color. The described methods range from simply painting an image to a flat surface to simulating depth using the idea of motion parallax.

The lesson learned, is that the introduction of programmable shader programs allows for a number of innovative surface reproduction methods that can simulate more complex appearances than just applying a color. Parallax mapping even resembles the goal of this work, as it can present a view-dependent surface appearance.

However, none of the above methods model the surface radiance in any view-dependent manner. Furthermore, all the techniques described above are based on the basic idea that each fragment is shaded according to a predefined shading model (e.g. Phong) and that light sources are modeled explicitly in the scene. It would be advantageous to alleviate these constraints, and allow for reconstruction of surfaces with more subtle details. It therefore seems feasible to strive towards recreating the view-dependent appearance of surfaces on a per-pixel level by implementing custom shader programs.

5 Design

This chapter presents the overall design of the developed framework for sampling, storing and recreating surface radiance. The design is derived from the processes and concepts described in the foregoing chapters. The design also acts as a basis for the problem statement presented in the forthcoming chapter.

5.1 The Task at Hand

Figure 5.1 presents the overall concept of the task at hand. A surface is to be sampled by taking a number of photographs of it from different directions. The surface is subdivided into many small patches, and the radiances sampled for each patch are saved. The sampling process is quite cumbersome, due to the amount of pictures that must be taken of the surface. This is naturally an offline process and is only done once for a given setup of the scene.

When having the radiances in many directions for each of the small patches of the surface, the task is then to reconstruct the view-dependent appearance of that surface. It must also be possible to present it in a real time computer graphics application.

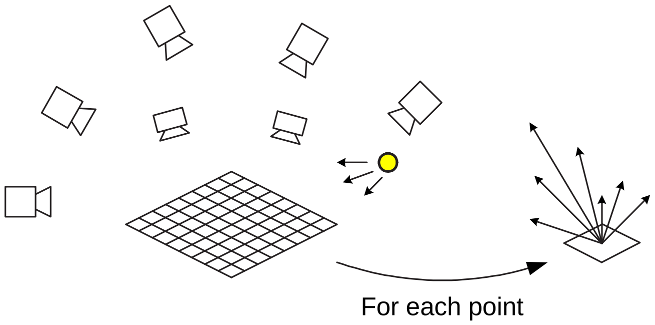


Figure 5.1: A figure of the flow in the framework. Images of a surface are acquired to capture the radiance distributions for each patch on the surface which can then be visualized.

Different ways of reconstructing real world surfaces in a view-dependent manner were analyzed in Chapter 3. The two main approaches were to either interpolate between a series of images or to fit the observed image data to a parametric model. It is wanted to be able to reconstruct subtle details in the surface that are visible only within small ranges of view directions. In the former approach this requires many images, which requires storage for undesirably much data. The latter approach, which fits a model to the data obtained for each of the small surface patches requires significantly less data. However, the question is always how well a model can be fitted to the data and how well a representation the end result constitutes. In this work, it is chosen to investigate how well real world surfaces can be reproduced by fitting models to the obtained radiance data, and thus use significantly less data to represent the view-dependent appearance of real world surfaces. The fitting of models to data can take very long time, since it has to be done for each patch. This is also an offline procedure, as it might easily take many hours to complete.

The concept of representing the surface with many small patches is well-suited for an texturing approach inline with those investigated in Chapter 4. It is desired to represent subtle details that vary on a per-point basis, and therefore an individual representation of each point on the surface is needed. A texture contains a representation of the appearance of each texel within it, e.g. as pixels in an image. Each texel is then mappable to a point on the surface and it is possible to interpolate between texels to obtain representation of surface points that fall between texel locations. Instead of image pixels, the texture could contain the parameters to the model fitted to the corresponding point on the surface. This way, means are provided for reconstructing the surface including its subtle details, by representing each of its points by an individual model saved within a texture. Many complex texturing methods can run in real time, and it is also expected to be able to present the results of this work in real time.

5.2 Design of the Framework

To comply with the above, the surface must be sampled in a manner which makes it possible to estimate model parameters. The first step is therefore to sample the radiances from all points on a surface from different directions. From these samples the parameters to a given model can be estimated and saved in a texture. Afterwards, the texture can be loaded into a graphics application and the view-dependent radiance of the surface can be reconstructed from the parameters.

In order to test the feasibility of this approach, a framework of tools is build to handle different tasks. The outline of the developed framework is illustrated in Figure 5.2. It is a coarse subdivision of the tasks involved in this thesis. It is divided into an offline and

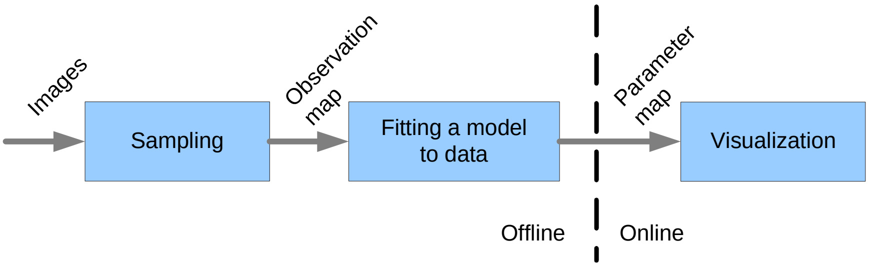
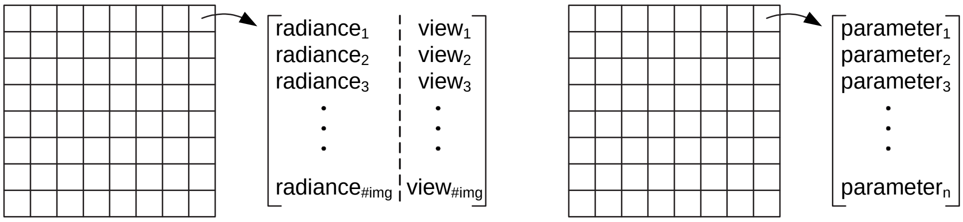


Figure 5.2: A figure of the flow in the framework. Images of a surface are sampled to estimate a model for the surface which can then be visualized.

an online part. The offline part covers sampling of the surface and fitting of the sampled data to a chosen model. The online part covers reconstruction of the surface at interactive frame rates.

Two new concepts are presented: "observation map" and "parameter map". These maps provide storage for the observed samples of the surface and the estimated model parameters, respectively. The content of the maps are outlined in Figure 5.3. The maps act as interfaces between the stages of the framework. A sampled point on the surface corresponds to a texel in each of the maps.

The observation map stores all observed radiances and their associated view direction for each point on the surface. The radiance is the R, G and B values recorded by the camera. The view direction is the spherical coordinate pair $[\theta, \phi]$[1] of the unit vector pointing towards the camera. The parameter map contains the parameters to the model for view-dependent radiance at each point.



(a) Observation map. Each texel contains all observed radiances and their associated view direction for a surface point.

(b) Parameter map. Each texel contains the model parameters for the corresponding surface point.

Figure 5.3: The content of the observation map and parameter map.

The flow of the work is summarized in the following steps, which relate to Figure 5.2:

- Through images of a surface, its radiances in a number of points are sampled into the observation map. This map contains all observations from all directions of all pixels comprising the surface. The direction to the camera is individually calculated for each point on the surface. The number of points comprising the surface depends on the chosen sampling resolution.
- The observation map is used to estimate the parameters for a model of the surface radiance at each sampled point. The estimated parameters are saved into the parameter map. This map contains the parameters to the model estimated for each point on the surface. The number of parameters in the map varies depending on the chosen model.
- Using the parameter map in a manner similar to texturing, the surface is visualized, seeking to recreate its real world radiance characteristics. The visualization application is the only part of the system that runs in real time, and it is interactive to allow the user to experience the view-dependency of the reconstructed surface.

[1] See Appendix A for information regarding spherical coordinates.

6 Problem Statement

This chapter presents the problem statement for the work in this thesis. On the basis of the preliminary analysis and the design, a number of goals are stipulated, each related to a stage in the framework. They also act as a measure for the thesis, when performing tests and evaluation at later point.

In the previous design chapter, a framework for testing the feasibility of the chosen approach was presented. The stages in the framework are to sample the surface, fit a parametric model to the observed data and finally reconstruct the surface appearance in a real time computer graphics application.

The aim of this thesis is to explore how the radiance of a surface can be sampled. Also, different models for representing the surface radiance must be explored. The chosen model should be able to cope with arbitrary radiance distribution from a surface point and represent it using few parameters. Finally, ways to efficiently reconstruct the surface from a parameter map in real time using modern graphics hardware must be found. These points can be summed up in the following problem statement:

Build a framework of tools to investigate the feasibility of sampling, modeling and recreating view-dependent radiance information from a surface for use in a real time application.

6.1 Goals

Based on the problem statement, a number of goals for the work in this thesis can be set up. They are formed on the basis of the stages in the framework presented in the design chapter. Thus, the following goals are stipulated for the work:

- Sample surfaces from different directions in a sufficient manner for later reconstruction.
- Capture the view-dependent characteristics of a surface and estimate parameters to models sufficient for describing them.
- Visualize the surface reconstructed at interactive frame rates in order to ensure that view-dependent properties of the surface have been captured and modeled correctly and sufficiently.

To verify that these goals are fulfilled a number of tests are performed. The tests include quantitative tests, where the estimated models are compared to the measured radiance data. Also more qualitative tests are performed, where the real and rendered surface is compared.

6.2 Delimitations

This thesis is a feasibility study with the goal of investigating methods for capturing and approximating the view-dependent radiance distribution of points on a real world surface. The thesis also includes examining recreation of the surfaces in real time computer graphics applications.

Sampling a surface with the intention of capturing its radiances is done by taking photographs of it. Photographs, and thus the sampled data, only represent the world as it looks at one given time instance. Therefore, in this work there is one overall delimitation. The world and all present objects, geometry and light sources are static. This way, it is a snapshot of the real world that is recorded and recreated.

No further explicit delimitations are made at this point. However, the right is reserved to delimit the scope of the work when having investigated aspects of it further.

6.3 Thesis Outline

Based on the design of the framework and the problem statement, this thesis is divided into the following parts. Each part analyzes and presents a solution to a step in the framework.

Sampling The sampling part analyzes how the surface radiance can be captured using a camera. Also, ways of choosing camera positions covering the hemisphere above the surface are explored. Finally, the part presents a method for extracting the observation map described in the design chapter from images of the surface.

Model Fitting The model fitting part analyzes different methods for representing the surface radiance. Two different models are chosen. The part ends with comparative tests of the two models.

Visualization In the visualization part, the modern GPU is analyzed to find means for reconstructing the surface appearance from the parameter map. Two different shader programs are implemented to reconstruct the surface appearance from both of the chosen parametric models.

Evaluation The evaluation part performs tests that compare reconstructed and real surfaces. Following the tests, the results are discussed before the work is concluded upon and topics for further studies are suggested.

Appendices A number of appendices describing subjects ranging from mathematical preliminaries over radiometry to camera calibration are presented in this last part. These serve as a supplement to the main report. References to the appendices are made when relevant.

Part II
Sampling

This part accounts for the process of sampling of points on the surface of a real world or synthetic object.

First, it is investigated how to correctly record the radiances from of real world surfaces by taking photographs of it. This is done by composing multiple shots into one image containing radiance values covering a large dynamic range. In order to present such High Dynamic Range (HDR) images on Low Dynamic Range (LDR) computer screens, a simple global tone mapping operator is chosen.

When having established a method for correctly recording real world radiances, it is discussed how a scheme for sampling a surface from view directions covering the hemisphere can be performed. Different schemes are presented and the problem of determining the camera position when taking real world photographs is outlined.

This issue is resolved by use of standard camera calibration techniques, which are used not only to determine camera position but also to map between points on the surface and image coordinates to obtain the corresponding radiance values.

Lastly, the three-channel radiance values and the directions to the camera are written into the observation map. In this map, a texel corresponds to one sample point on the surface.

High Dynamic Range Imaging 7

This chapter investigates the term known as High Dynamic Range (HDR) imaging, which is used when sampling real surfaces in this thesis. The chapter starts by describing one problem with general imaging, namely that it is not readily possible to obtain images where details are present in both dark and bright regions simultaneously, a task requiring use of different exposure times. A method for constructing HDR images is described, as well as examples of methods for mapping them into image values that can be viewed on normal computer screens are given.

7.1 The Need for High Dynamic Range in Imaging

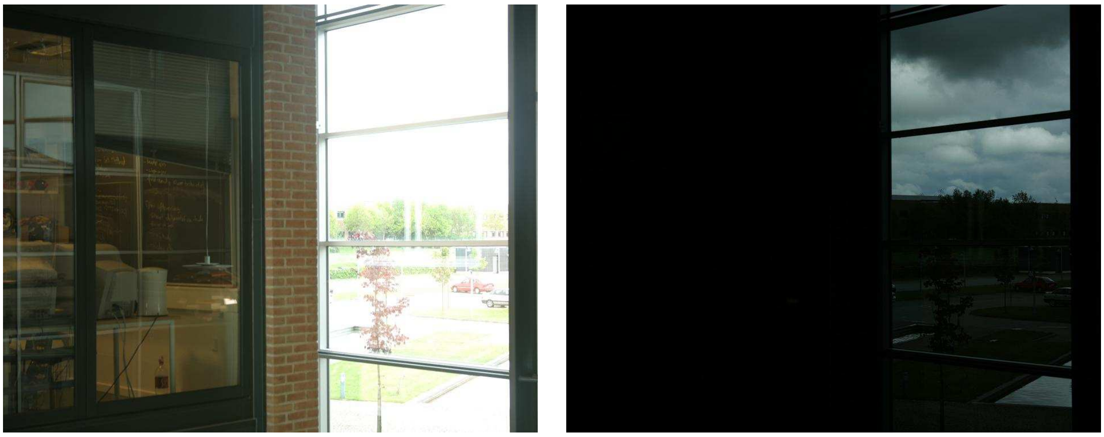
When sampling a surface, its radiance in a number of points is what is of interest. Real radiance has a very high dynamic range, so in order to be able to sample these radiances correctly using cameras, it is required to use high dynamic range imaging techniques.

The basic functionality of cameras is to measure radiance[1], which is defined as flux passing through, leaving or arriving at a point in a particular direction [RWPD06]. For this purpose, cameras are equipped with light-sensitive chips. When taking pictures with a digital camera, a number of settings must be set correctly for the resulting picture to show the desired output. If the scene is very dark, e.g. taken at night, the exposure time must be relatively long to let enough light into the light sensitive chip. Reversely, if the image is taken at noon outside on a sunny day, the exposure time should be relatively short due to the large amount of light.

Figure 7.1(a) shows an image of an indoor scene taken with a relatively long exposure time. Details are visible in the left part of the image, but the right part of the image showing the glass facade is clearly overexposed, and all pixels are either fully or nearly saturated. Figure 7.1(b) shows a picture of the same scene, taken in immediate succession. However, the exposure time has been shortened to avoid saturation of pixels showing the sky. Now the details are discernable in the outdoor part of the picture, but the indoor part of the scene is not at all visible because it is too dark.

The above described phenomenon is due to the fact that cameras can only record radiance within a low dynamic range in one image. Natural radiance is linear, and has a large dynamic range from the small amount of light in shadowy areas to the immense amount of light coming from the sky or the sun. The human visual system can adapt to such large dynamic ranges as the order of 10^{10} [GW02]. However it can only adapt to a smaller range of it at the time and requires some time for adaptation. This can e.g. be experienced when

[1] See Appendix C for a description of radiometry and light in general.



(a) Long exposure time.

(b) Short exposure time.

Figure 7.1: Two images taken in immediate succession. Long exposure time enables details in dark areas while bright areas are saturated. Short exposure time enables inspection of details in bright regions, but darker regions are not visible.

walking from the indoors to the outdoors on a sunny day. Furthermore, experiments have shown that the perceived intensity of the human visual system is a logarithmic function of the radiance incident on the eye [GW02]. As a result of this, image encoding techniques apply gamma compression to account for this and to optimize perceptual performance. Monitors then apply gamma expansion to obtain a linear form of the image signal for presentation.

7.2 Constructing High Dynamic Range Images

To alleviate the restraint in dynamic range of normal LDR images, HDR imaging was invented. A method for combining multiple exposures of a scene into a single HDR image is presented in [DM97], where a response curve for a specific imaging process is found and used for recovering relative radiances from a number of images. [DM97] is the main source for the following sections.

7.2.1 Recovering the Response Curve for an Imaging Process

The method exploits the reciprocity of both photochemical and electronic imaging systems. For a pixel in an image, the exposure (not to be confused with exposure *time*) is defined as $X = E \cdot \Delta t$ (radiance times exposure time). In the following, radiance is the amount of light that enters a camera into a certain pixel, i.e. the light that can enter through the pyramidal volume that the pixel corners form with the camera center.

The above is valid under the assumption that only the product $E \cdot \Delta t$ is important for the result (i.e. halving E and doubling Δt would not change the resulting X). The end result of an imaging process, be it digital where the camera maps radiance values to image values or photochemical with development of films and producing images from negatives or scanning of them, is a digital number, Z. This Z value is a found through a non-linear mapping of the original pixel exposure X, a function $Z = f(X)$.

By assuming that $f(X)$ is smooth and monotonically increasing, its inverse, $X = f^{-1}(Z)$ can be found. $f^{-1}()$ is the response curve. When being able to calculate X from an image value, the radiance can be found as $E = \frac{X}{\Delta t}$, when the exposure time is known. To find the radiances, E_i, and the function $f^{-1}()$, a set of functions are derived as:

$$\begin{aligned} f^{-1}(Z_{ij}) &= X_{ij} \\ &= E_i \cdot \Delta t_j \\ \Rightarrow \ln(f^{-1}(Z_{ij})) &= \ln(E_i) + \ln(\Delta t_j) \end{aligned} \tag{7.1}$$

where i ranges over pixels and j ranges over exposure times (photographs). The Z_{ij} and Δt values are known. This set of equations can then be used to minimize a quadratic function using straightforward linear least squares. $\ln(f^{-1}(Z))$ is then recovered as a finite number of values that it can attain by use of a representative number of pixel locations in the images and a number of photographs that cover the dynamic range of the scene.

The response curve is found for an entire image process. A recovered response curve thus describes how one particular digital camera converts radiances to image values. In principle, two images with sufficiently many different radiance values and similar enough exposure times for their contribution to the response function to overlap can be used to obtain a response curve for an imaging process. However, increasing the amount of images with differing exposure times obviously improves performance wrt. noise sensitivity.

7.2.2 Converting to Relative Radiance Values

When the response curve for the employed imaging process is recovered, pixel values in images with known exposure time can be converted to relative radiance values. To get the best result, all available exposures for a pixel are used to compute its radiance, where a weighting function, $w(Z_{ij})$, tones down the influence of pixels close to being over- or underexposed. Thus, the radiance can be obtained by rearraging Equation 7.1, summing over all exposures and applying the weighting function:

$$\begin{aligned} \ln(E_i) &= \ln(f^{-1}(Z_{ij})) - \ln(\Delta t_j) \\ &= \frac{\sum_{j=1}^{P} w(Z_{ij})(\ln(f^{-1}(Z_{ij})) - \ln(\Delta t_j))}{\sum_{j=1}^{P} w(Z_{ij})} \end{aligned}$$

where i again ranges over pixels, and P is the number of exposures. Combining multiple exposures when finding the relative radiance in a pixel reduces noise in recovered radiance values.

The number of images needed to recover the relative radiance value of a pixel depends on the dynamic range of the scene. To ensure that all parts of the scene are photographed with an exposure that puts them in the working range of the response curve (the middle range), $\lceil \frac{R}{F} \rceil$ images are needed. R is the range in which it is desired to recover radiances accurately, and F is the dynamic range that the camera can represent within its working range. This means, that relative radiance values for a scene can be retrieved with one image, as long as the dynamic range of the scene is representable within the working range of the camera used to record it.

To obtain the absolute radiance values, a calibration light source of known radiance can be photographed and used to scale the HDR image radiance values. The above paragraphs also implicitly treated images as single channel images, but the process is repeatable for multiple channels to obtain three independent relative radiance maps for an RGB image.

It should be noted that global illumination solutions also produce radiance values. Therefore a need for handling radiance values instead of RGB-values does not only arise when wanting to sample the real world correctly, but also when simulating it using a global illumination solution. Thus, the following description of how HDR images are viewed is applicable both to real world and synthethic HDR images.

7.3 Viewing High Dynamic Range Images

It is expected that future image recording and viewing equipment will be able to show and handle high dynamic range scenes directly [SAG$^+$05] and all image radiance values are handled as floating point values. However, at the current time typically only 8 bits are used per color channel in digital images. Furthermore, there is also some importance in the monitor the images are viewed on. Even though many of the newest, high-end LCD or plasma screens might have contrast ratios of up to 6000:1, they still only have 8 bits per color channels to display colors within this range. Both these facts limit the available dynamic range for presenting images. Thus, the need for tone mapping operators is evident. A tone mapping operator, transforms the radiance values stored in an HDR image, allowing for it to be viewed on an LDR screen.

7.3.1 Tone Mapping Operators

The overall task of tone mapping operators is to map the radiance values of an HDR image into brightness values in an image for presentation on normal computer screens. Tone mapping operators can be classified in three ways: whether they are local or global, whether they work in the spatial or frequency domain and according to which real world phenomenon they seek to preserve in the output image. For brevity, only the difference that lies within an operator being local and global is touched upon in the following. An extensive walkthrough and description of different operators is given in [RWPD06].

Local operators work on some neighborhood of the current pixel in question, which then forms a unique operator for that pixel. Local operators thus treat bright pixels in dark neighborhoods differently from bright pixels in bright neighborhoods. The same is valid for dark pixels in dark and bright neighbordhoods. Due to the fact that a unique operator must be calculated on a per-pixel basis, local operators are typically not well-suited for real time rendering.

An example of a locally tone mapped HDR image composed with Photomatix is shown in Figure 7.2. The left part of the figure shows the original images used to make the HDR image and their respective exposure times. The resulting HDR image is tone mapped with the built-in local tone mapping functionality of Photomatix to present an image with details in both dark and bright regions simultaneously.

The difference between local and global operators is which neighborhood in the image they take into consideration. Global operators can take the entire image into consideration and apply the exact same operator to each pixel to e.g. preserve the perceived brightness or contrast in the image. They can also be a simple as applying e.g. the same sigmoid or logarithmic function to each pixel, which provides for quick tone mapping suitable for real time rendering.

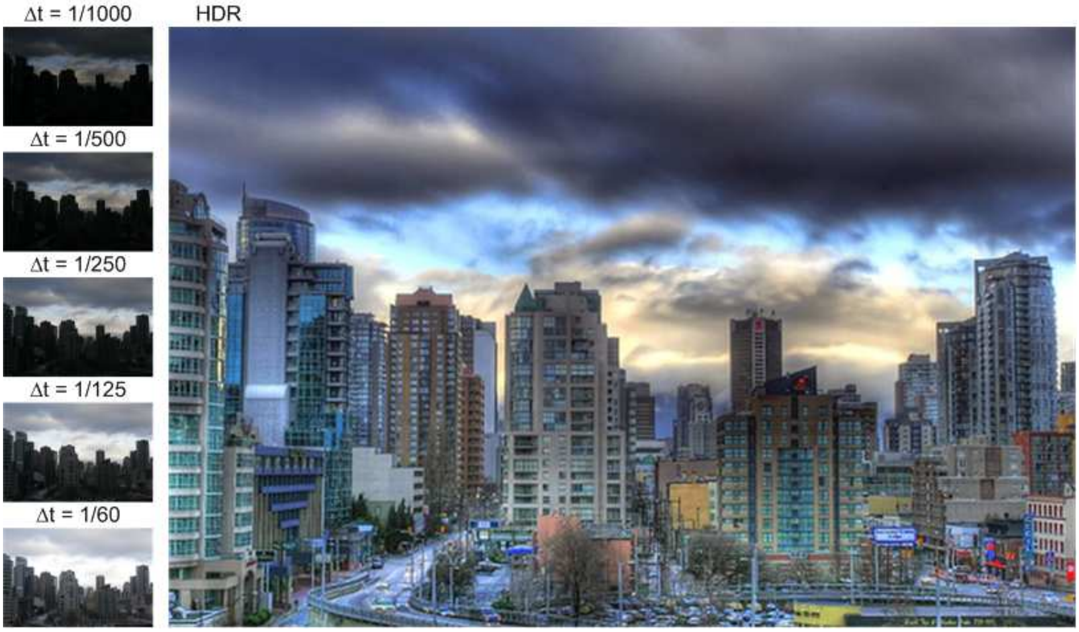


Figure 7.2: An example of a tone mapped HDR image. The five images used to construct the HDR image are shown to the left with their respective exposure time stated above. The HDR image is constructed using Photomatix, and a local tone mapping operator is applied to produce a visually pleasing image with details in both bright and dark areas simultaneously. Images from [Sto].

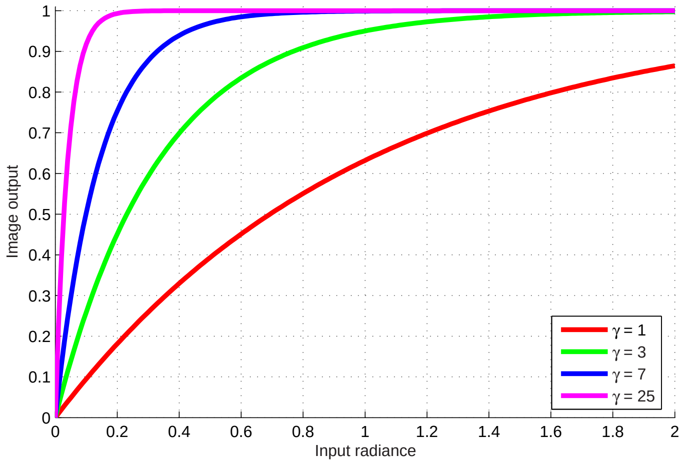


Figure 7.3: A simple tone mapping function, $y = 1 - \exp(-\gamma \cdot x)$.

7.3.2 The Exposure Function

A simple global tone mapping operator working on individual pixels is presented in [Eli07]. It is based on the idea of chemicals in light sensitive film breaking down when exposed to light. The reverse of the rate at which chemicals break down as a function of light, gives a description of brightness as a function of incoming light, i.e. incoming radiance. The amount of incoming light in a camera can be controlled with either the exposure time or the width of the aperture. This tone mapping operator produces a brightness on the basis on a user-controlled γ values that indicate the exposure time. Figure 7.3 shows this tone mapping operator for different values of γ. The function is $y = 1 - \exp(-\gamma \cdot x)$. The larger the γ value, the brighter the output image. The asymptote at 1 ensures no overflow in output images.

Using this simple tone mapping operator, the user is not enforced to apply some overall preservation technique but can finely tune the γ parameter to show details in any desired region of the image.

7.4 Summary

It has been established that, if the true radiances of real world surfaces are to be sampled correctly, it is necessary to take multiple pictures from the same sample direction with different exposure times and assemble them into an HDR image. As these HDR images can not be shown in full on a normal computer screen, the simple user controlled “Exposure Function” global tone mapping operator is employed in this work. The user can investigate any chosen part of the dynamic range of the reconstructed surface by adjusting γ. The chosen tone mapping procedure resembles the way that the human vision adjusts itself to be able to view details in either dark or bright ranges.

8 Sampling over a Hemisphere

This chapter presents the concept of sampling a surface. Different schemes for covering the hemisphere over the surface are presented and discussed. They act as inspiration to the following parts. Differences between sampling real world surfaces and synthetic surfaces is also briefly touched upon.

Generally, when deciding how to sample a signal, the sampling frequency is determined on the basis of the frequency of the signal. The Nyquist sampling theorem states that the sampling rate must be larger than two times the highest frequency of the signal, to uniquely determine a signal by its samples and avoid aliasing [OS89]. Carrying this over to sampling the radiance distribution from a point, which is defined over the entire hemisphere, the signal frequency is the frequency with which the radiance from the point changes over the hemisphere. Radiance can vary due to surface properties as well as the amount and placement of light sources.

In related work, the amount of samples on the hemisphere ranges from relatively few in the VDTM work [DBY98] over 205 for isotropic and 410 for anisotropic surfaces in the work regarding BTFs [DvGNK97] to more than 500 in [SHR$^+$99], where BRDFs are reconstructed through images of the surface.

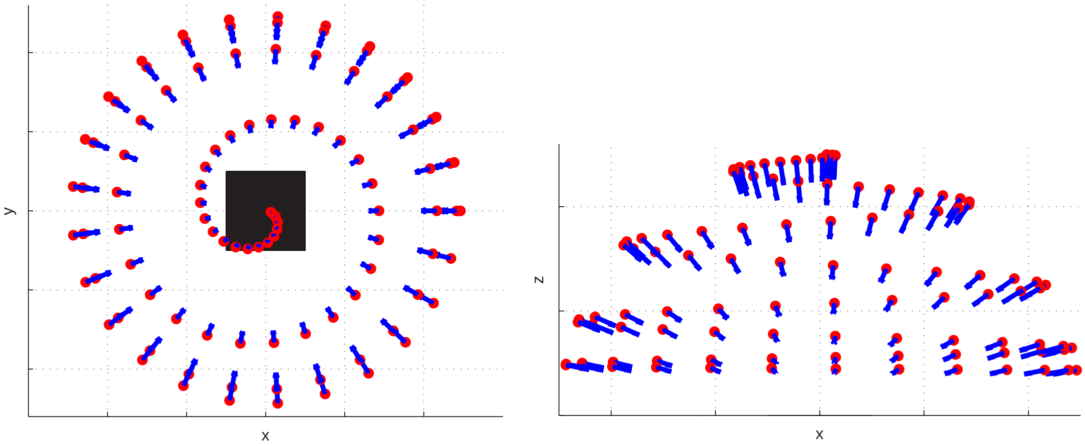
The following sections present examples of different overall schemes for sampling a surface from points on the hemisphere over it. The chapter is ended with a discussion of how these are applicable when sampling real world data.

8.1 Sampling Schemes

The frequency of the radiant exitance from a point on a real world surface is hard to determine precisely without sampling it very densely, as many factors have influence on it. Therefore, the sampling is prone to becoming a case of trial-and-error, where an initiating guess forms the basis for the sampling rate. The most prevalent concern is typically to cover the entire hemisphere. If the results are unacceptable, the sampling can be augmented with more samples, until acceptable results are achieved. The following describes three different overall schemes for sampling.

8.1.1 Spiral sampling

The concept of sampling the hemisphere in a spiral is exemplified in Figure 8.1. The basic idea of this non-uniform sampling scheme is, that an overall number of desired samples, N, is indicated along with how many times the hemisphere should be circled, R. The zenith has values within the range $[\pi; 0]$ with $\frac{\pi}{N}$ decrements and the azimuth attains values in the range $[0; 2\pi \cdot R[$ with increments of $\frac{2\pi \cdot R}{N}$. Problems with unnecessary oversampling in the direction of the surface normal at the point can occur because of the small distance between sampling points. This can be seen in the middle of Figure 8.1(a). The issue can be alleviated by letting the zenith go from π to 0 with non-equal decrements.



(a) Spiral sampling, xy-view.

(b) Spiral sampling, xz-view.

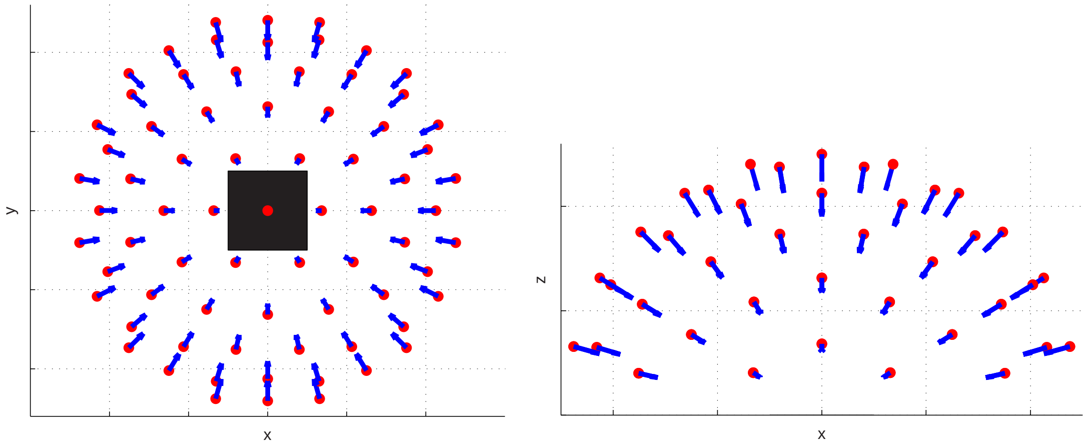
Figure 8.1: An example of sampling the hemisphere by moving around and upwards, resulting in a spiral.

8.1.2 Icosahedron sampling

Another approach to sampling the hemisphere is by taking a starting point in the structure of a regular icosahedron. A regular icosahedron has 20 faces and 12 vertices, which are equilateral triangles. Each vertex then constitutes a sampling point. For higher sampling frequency, the edges of the icosahedron can be divided into two by making a new vertex at the edge midpoint and projecting this onto the unit sphere, a task known as mesh refinement. This sampling scheme is uniform with relation to distance between neighboring sampling points, and thus with relation to how large an area of the hemisphere they each cover. Figure 8.2 shows an example of sampling points scattered by the icosahedron principle, with a total of 73 sampling points.

8.1.3 Latitude-longitude sampling

Latitude and longitude are normally used to indicate position on the earth. Latitude indicates the vertical position and longitude indicates horizontal position. This is the idea that inspired to the latitude-longitude sampling scheme exemplified in Figure 8.3. Here, a number of desired zenith values are predefined, as well as how many samples to take around the circle for each zenith value. In the figure, the zenith varied from $[\frac{\pi}{7}; \frac{6\pi}{7}]$ with $\frac{\pi}{7}$

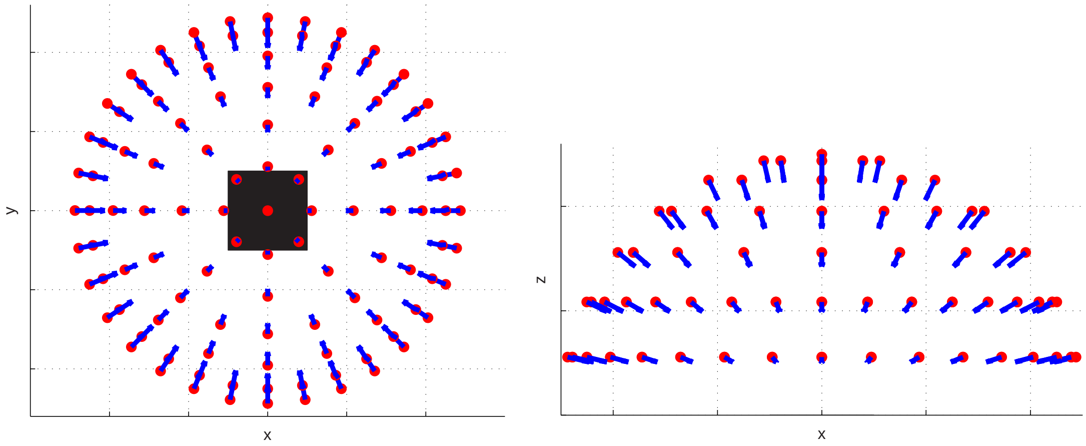


(a) Icosahedron sampling, xy-view.

(b) Icosahedron sampling, xz-view.

Figure 8.2: Sampling a hemisphere on the basis of an icosahedron, dividing the hemisphere into equal areas. The sampling points are the equidistant vertices of the icosahedron mesh.

increments. The number of azimuth values is 32 for the two lower bands, 16 for the middle bands and 8 for the two upper bands. A single, extra sampling point in the surface normal direction is added, resulting in 113 samples. These values can of course be changed, to fit the desired amount of samples, but it should be kept in mind to sample more in the lower bands, where the circumference is larger. This could be accommodated by controlling the amount of azimuthal samples with a sine function of the zenith.



(a) Latitude-longitude sampling, xy-view.

(b) Latitude-longitude sampling, xz-view.

Figure 8.3: Sampling of a hemisphere at fixed zenith values, with azimuth values in the range $[0; 2\pi[$.

8.2 Sampling Real World Surfaces

All of the above described sampling schemes are addressed at sampling synthetic data. This has one simple reason; it is very hard to precisely place the camera at a certain sampling point and with a certain direction. Therefore, it seems reasonable to strive to follow a sampling schemes as good as possible, and take into consideration that the desired sampling placement and direction is not fully achievable. Therefore, there is a need for means for calculating the actual camera position and the direction to each of the sampling points on the surface. This is addressed in the following chapter.

In all of the above sample schemes, it is assumed that the camera is always the exact same distance from the center of the sampled surface. This is however not a requirement. As accounted for in [SAG$^+$05], radiance does not vary along a line in space, and thus, surface radiance does not change as a function of distance between camera and surface. This is because the area of the surface covered by some solid angle is proportional to squared distance and light reaching the camera is inversely proportional to the same area, which cancels out distance as a factor in the final result.

The icosahedron sampling scheme is the only uniform one of the above. Uniform sampling ensures that all view directions are given equal importance. Another uniform sampling scheme, employed in [Gre03], is stratified sampling. In this approach, unbiased random samples are obtained by dividing a $[0..1, 0..1]$ space into N by N sample cells and randomly sample from a direction within each of these cells. These sample points are then converted to (θ, ϕ) angle space.

The other two sampling schemes are non-uniform over the hemisphere and this might be disadvantageous, due to the uneven areas covered by the samples. This, however, does not mean that a sampling scheme should always be followed. In some cases there might be directions that require more attention than others, e.g. highlights, and deliberate non-uniform sampling can be a valid approach. Thus, when photographing a real world surface it might be a sufficient approach to sample from random, but still well-considered, directions.

8.3 Summary

A number of different sampling schemes has been investigated and the problem of sampling real world surface is discussed. However, no test of their feasibility has been performed, as this is better evaluated in retrospect when a model has been fitted to the data sampled with any of the schemes. It is evident, that it is not possible to precisely place a camera in the positions that a sampling scheme dictates. Therefore it is decided to follow a suitable sampling scheme, place the camera as close to the desired position as possible and investigate means for calculating the actual position of a camera.

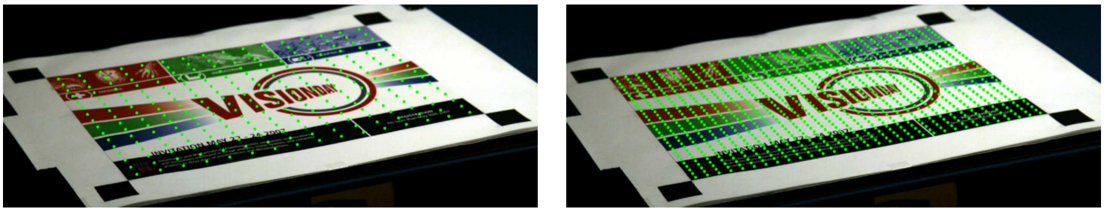
9 Constructing an Observation Map

In this chapter, it is described how the position of each of the sampling points on the surface are found, as well as how the camera position relative to the surface is found. It is also described how the direction from each of the sample points to the camera position for each image is found, before the contents of the observation map are accounted for.

9.1 Sampling Resolution

As mentioned in the end of the previous chapter, there is a need for determining the camera position relative to the surface and the direction from all sample points on the surface to the camera. This also implies actually calculating the location of each of the sample points.

At first, the number of sample points on the surface must be determined. This decision is based on the frequency the texture of the surface exhibits. The more the color or reflection properties of the surface changes, the higher a resolution is needed. If text is present on the surface, it typically requires a high resolution whereas the surface of a table might not vary that much and could suffice with a low resolution. The resolution with which the surface is sampled is directly transferred to the resolution of the observation map, as each texel herein contains the sampled information for one point on the surface. Obviously, the resolution should fit the size and form of the patch of the surface. Figure 9.1 shows an example of using different sampling resolutions.



(a) Resolution = 16x16.

(b) Resolution = 32x32.

Figure 9.1: Two different sampling resolutions. They are both set much to low, but serves illustrative purposes. To be able to reconstruct the text and detailed figures on the surface, a much higher sampling resolution is obviously needed.

9.2 Deriving Radiance from Surface Sample Points

When determining a unique radiance value in a direction for each sample point on the surface, it is necessary to be able to transform the points into a position in the image plane. This is possible when the intrinsic and extrinsic camera parameters are known. In this thesis, these are estimated using the camera calibration[1] toolbox by [Bou07].

The methods in the toolbox rely on rectangle corners to estimate the camera parameters. Normally a checkerboard-pattern is used to provide a large number of corners. However, there are generally no corners on the sampled surface. Therefore, the corners of the surface itself are used to estimate the extrinsic camera parameters for each view. As the corners of the surface are to be used in the calibration process, two restrictions are imposed on the possible surfaces that can be sampled. It must be rectangular and the corners of it must be visible in all views. A standard checkerboard calibration object is still used to initially estimate the intrinsic parameters.

As the real world size of the surface patch is known, the resolution has been predetermined and the real world coordinate system can be defined as desired, it is possible to readily map between position in observation map and position on the surface. Figure 9.2 exemplifies this. In this example the observation map consists of four texels as $S = T = 2$. The size of the surface is A4 (width = 297mm, height = 210mm).

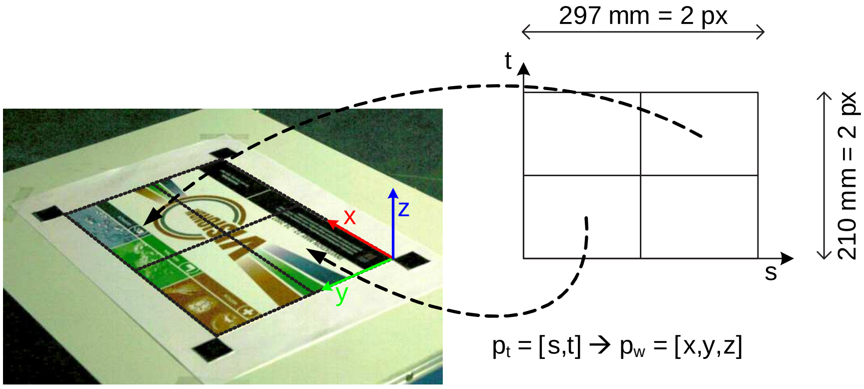


Figure 9.2: The mapping between observation map position and real world position is readily possible with known surface size, real world position and observation map resolution.

A zero-indexed texel position in this case $\mathbf{p}_t = [s, t]$ can be converted to real world coordinates $\mathbf{p}_w = [x, y, z]$ by the following formula (the z-coordinate is by definition always 0):

$$x = \frac{s \cdot width}{S} + \frac{width}{2 \cdot S} \quad , \quad y = \frac{t \cdot width}{T} + \frac{width}{2 \cdot T} \quad \text{and} \quad z = 0$$

This produces, for the arrows in Figure 9.2, the following mappings:

$$[0, 0] \rightarrow [74.25, 52.5, 0.0] \quad \text{and} \quad [1, 1] \rightarrow [222.75, 157.5, 0.0]$$

[1] See Appendix F for a general description of camera calibration.

The mapping between camera and world coordinates is given by the extrinsic camera parameters. The extrinsic parameters are calculated on a per-image basis and consists of a rotation matrix and translation vector. The following equation calculates the camera coordinates from a point in world coordinates:

$$\mathbf{p}_c = \mathbf{R} \cdot \mathbf{p}_w + \mathbf{t} \tag{9.1}$$

The right part of Figure 9.3 shows a point on the surface (in both world and camera coordinates). The camera and image plane are shown in the left part of the figure.

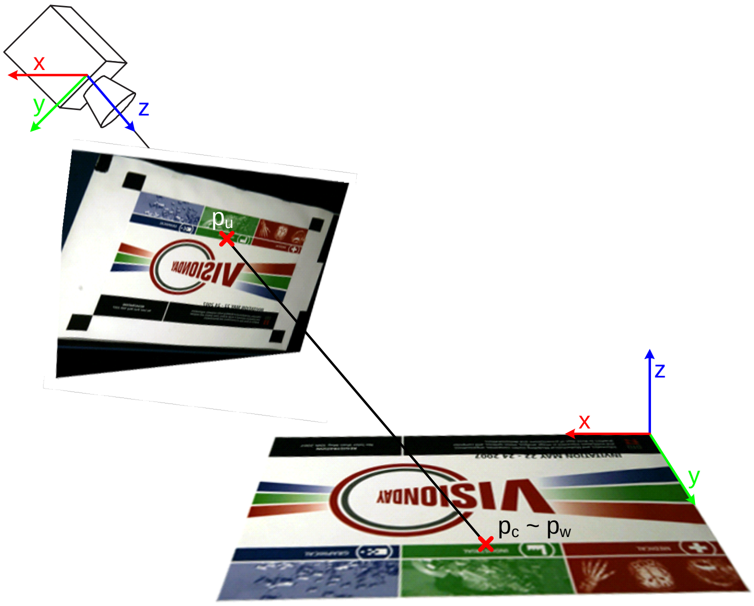


Figure 9.3: Mapping from real world coordinate system position over camera coordinate system position to undistorted image coordinates is possible using the tools in the camera calibration toolbox from [Bou07].

Figure 9.3 shows that the undistorted image coordinate, $\mathbf{p}_u(x, y)$, is found directly by projective tranformation. To obtain the actual point in the image, it is necessary to compensate for lens distortion, which is modeled by the intrinsic camera parameters. These parameters comprise focal length, radial and tangential distortion, physical dimensions of pixels and the principal point coordinates.

The radiance for the given sample point, can then be found by bilinear interpolation in the recorded image at the resulting distorted image point, $\mathbf{p}_d(x, y)$.

9.3 Deriving Camera Direction at Sample Points

The direction from a sample point to the camera can be found as a vector between the two positions in the 3D world coordinate space. The world coordinates for the sample point are found at an early stage in the procedure described in the previous section. The world coordinates for the camera are found by use of the extrinsic camera parameters. The camera position in its own coordinate system is by definition $\mathbf{p}_{c,cam} = [0, 0, 0]$, this can be inserted into Equation 9.1 which is rearranged to:

$$\mathbf{p}_{w,cam} = \mathbf{R}^T \cdot (\mathbf{p}_{c,cam} - \mathbf{t}) = \mathbf{R}^T \cdot -\mathbf{t}$$

With this information calculated for each sample point from each view direction, it is now possible to put the information into an observation map according to the form specified in Chapter 5.

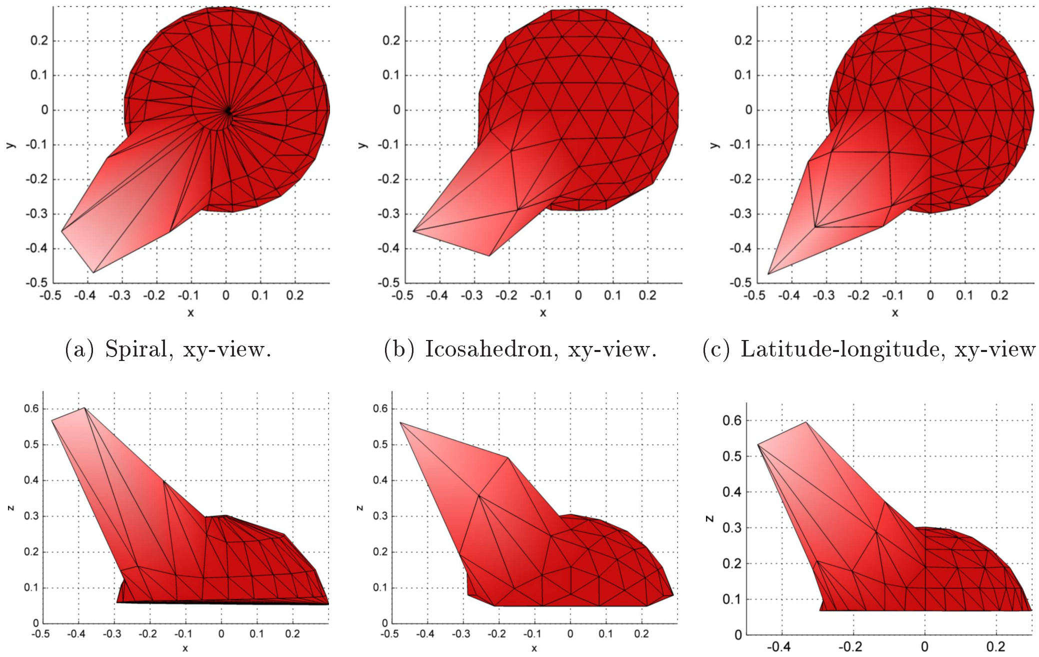
9.4 Exploring the Content of an Observation Map

As explained, the content of the observation map is a texel containing all recorded radiances and their corresponding direction for each sample point on the surface.

Figure 9.4 shows plots of the data contained within a texel of an observation map for a surface sampled using the different sampling schemes. In all plots, a mesh grid is overlaid the vertices. Each vertex constitutes a unique sample direction, and its length from the origin indicates intensity. The color of a vertex is determined by the perceived color of sample point from the given direction. The sample point is located in the origin. The top row shows the mesh seen from above and the bottom row shows the mesh from the side. All schemes detect the highlight that is present, however the form of it varies from scheme to scheme. In the figures, the number of sample directions for a scheme corresponds to the number mentioned in the previous chapter. If a higher number of sample directions were used, a more smooth representation of the highlight would naturally be present. It is a deliberate choice to show an example with a clear highlight, because there would be no difference in use of sampling scheme for a predominantly diffuse radiance characteristic.

The observation map contains much information, and it is informative to look at what kind of sizes such maps actually attain. The direction and corresponding RGB-values are saved for each view direction per texel. This is two floats for direction and three floats for color. This is saved for all view directions, N, and for all texels in the map, $S \cdot T$. In total, this becomes $S \cdot T \cdot N \cdot 5$ floats.

Table 9.1 lists a number of sizes of observation maps and their corresponding size in megabytes. The first row lists the width and height of the map, and the following rows list amount of megabytes needed to store the map with $N = 100$ and 300. It is evident that for sizes of 512x512 and above, the storage requirements are much too high for real time applications. This resolution might even be less than what is necessary for high frequent surfaces containing e.g. text. These storage requirements resemble those that would be needed if image interpolation were used, with no compression. The model representation fitted to the data should lie very much below these storage requirements.



(a) Spiral, xy-view.

(b) Icosahedron, xy-view.

(c) Latitude-longitude, xy-view.

(d) Spiral, xz-view.

(e) Icosahedron, xz-view.

(f) Latitude-longitude, xz-view.

Figure 9.4: Visualization of one texel in the observation maps obtained with the described sampling schemes.

SxT	**32x32**	**64x64**	**128x128**	**256x256**	**512x512**	**1024x1024**
N=100	1.95	7.81	31.25	125	500	2000
N=300	5.86	23.44	93.75	375	1500	6000

Table 9.1: The sizes in megabytes of differently sized observation maps, with differing amount of view directions. It is clear that storage consumption rapidly gets very high.

9.5 Summary

In this chapter, it was decided to place the camera as close as possible to the desired position and employ camera calibration techniques to determine the actual position of a camera. The camera calibration technique has been accounted for in this chapter. It also provided means for calculating positions of all sample points on the surface in the images, as well as a unique direction to the camera for each point. This is done with the imposed delimitation that the surface must be rectangular and entirely visible in all images. The contents of the observation map has also been accounted for, and example figures provide a visualization of what data is actually stored in the map and what it represents.

Part III
Model Fitting

This part accounts for how the radiance values for each texel in the observation map can be represented by a parametric model.

First, it is analyzed which models can be used to represent the radiance distributions from a sample point. Inspirations are sought from empirical shading models used in computer graphics, but more general models are also investigated. It is chosen to use both a Phong reflection inspired model and spherical harmonics (SH) to represent the radiance distribution for a texel in the observation map.

For the Phong model, it is described how this can be modified to model the captured radiance distribution, instead of material properties. The parameters for the model are estimated using a non-linear optimization technique.

The theory behind SH is presented and it is described how the SH parameters can be found by projecting the measured radiance distribution onto the SH basis functions. Two different approaches for handling the fact that surface points are only sampled from the hemisphere is presented and tested.

Lastly, the Phong and SH models are compared in different scenarios. It is investigated when to use which model. Also, different sampling strategies are tested to investigate the number of images needed to estimate the model parameters accurately.

10 Models for Point Radiances

This chapter analyzes different models for the captured radiance distribution from a sample point. First, empirical models are considered. These are well known in computer graphics. Next, some more general models are examined which can represent arbitrary radiance distributions.

In the design chapter, it was decided that a model should be fitted to the captured radiance distribution. An example of the radiance distribution for a sample point is plotted in Figure 10.1.

The radiance distribution from a single point can vary a lot. In the figure, it is apparent that the surface is lit by a single light source because of the strong reflected radiance in one particular direction. The complexity of the radiance distribution dependents on the complexity of light in the environment and the properties of the surface material.

The requirement for a model for the radiance distribution is that it should be able to represent the radiance distribution compactly, i.e. by use of few parameters. Given a view direction and the model parameters, the radiance in the view direction must be reconstructed efficiently, since it has to be done in real time.

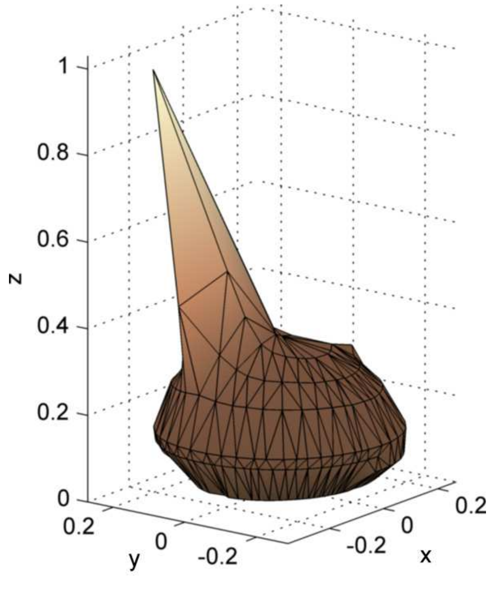


Figure 10.1: Radiance distribution of a sample point

In the following two sections, two different classes of possible models are presented. First, some well known empirical models for material properties in computer graphics are analyzed. Even though it is not wanted to estimate the material properties but rather the reflected radiances, these models can with some assumptions about the lighting environment be modified to model a radiance distribution instead of material properties. Next, some more general methods are presented. These are in theory able to model the reflected radiances in more arbitrary lighting environments.

10.1 Empirical Models

Many different shading models are used in computer graphics for modeling different materials. These models represent the BRDF of the material. The BRDF is essentially a 4D function which estimates the reflected radiance from a point in a given direction (usually towards the eye) given an incident light[1]. The complexity of the models vary greatly ranging from relative simple models giving acceptable results to more advance and physically correct models.

The radiance distribution from a sample point only depends on the view direction and not on the direction to the light source. In order to use the BRDF models to fit the radiance distribution instead of the material properties, it must be reduced to a 2D function. This can be done by making assumptions about the lighting environment.

One example of work where shading models are fitted to measured data is [SHR$^+$99]. They attempt to iteratively fit simple shading models to the appearance of real-world surfaces using a series of photographs of that surface. The shading models employed are the Phong [Pho75], the Blinn-Phong [BN76] and the generalized cosine lobe [LFTG97] models and the sampling is performed under assumptions of known geometry, light source information, calibration grid and a homogenous reflection functions. Another example is [LKG$^+$01], which also seeks to fit measured surface data to the Lafortune model.

One of the most often used and well known shading models is the Phong reflection model. Phong proposed to model the intensity of the specular highlight from a point light source as the cosine of the angle between the view direction and the mirror reflection direction raised to the power of a shininess coefficient. An infinite shininess corresponds to a mirror, where the highlight is only seen from one single view direction, because the Phong model assumes point light sources. The complete shading model is a combination of the specular, diffuse and ambient term.

A modification to the Phong reflection model is the Blinn-Phong reflection model. This shading model is the one used in the fixed pipeline of modern graphics cards. Basically, it is a fast approximation to the Phong specular term. Instead of having to compute the mirror reflection direction, the half way vector between the view and lighting direction are used. The specular term is when estimated as the cosine of the angle between the half

[1] See Appendix D for information about the BRDF.

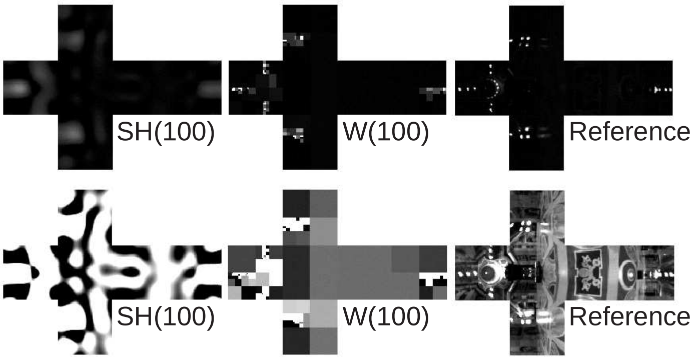


Figure 10.2: A comparison between two environment maps modeled using both spherical harmonics and spherical wavelets. Figure from [NRH03].

vector and the surface normal raised to the power of the shininess.

Other, more advanced models exist which try to model different phenomena and different material types. These have more parameters which

10.2 General Models

In order to circumvent the need for a known lighting environment, a more general representation is needed. Using a general approach, a much more arbitrary environment can be used. This might include e.g. an unknown number of light sources and area light sources.

An example of work seeking to fit the BRDF of complex surfaces to more general representations is [WAT92]. They use the SH basis to represented the BRDF of highly complex surfaces. The SH is a complete set of smooth basis functions defined over the sphere.

Another interesting work is performed by [NRH03], in which the lighting environment map and precomputed transfer functions are represented by a spherical wavelet basis. Using the wavelet basis more high-frequent details can be preserved. This is illustrated in Figure 10.2, where two different environment maps are represented by both a spherical harmonic and wavelet basis using 100 parameters. The disadvantage of the wavelet representation is that the representation is not as smooth as the spherical harmonics basis.

10.3 Summary

The above sections present two different overall approaches for representing the radiance distribution from a sampled point with a parametric model. The empirical models are compact, but require some knowledge about the number of light sources in the scene. Furthermore, the lights need to be approximately point light sources. Reversely, more general methods such as spherical harmonics and wavelets require no a priori knowledge, but require more parameters.

In this thesis, it is sought to represent the radiance from points on a surface in a very general lighting environment. Therefore it is natural to choose a general model. In the recent years in computer graphics, the SH basis has received a lot of attention and is well studied. Due to this, this thesis uses the SH basis functions as a representation for the radiance distribution.

However, also the possibility of representing the radiance distribution using a simple shading model is explored due to its compactness. The Phong reflection model is chosen for this purpose, because its parameters are quite intuitive and understandable.

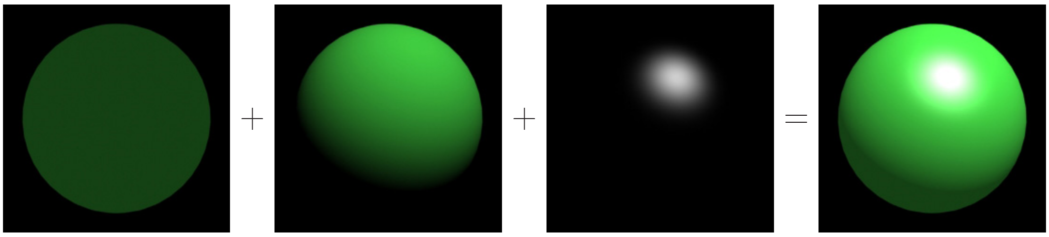
The following two chapters each present one of the two chosen models. They present the theory behind the models, how the parameters for each of them can be estimated and the test of them. Finally, a chapter compares the two models in different scenarios.

11 Phong Model

This chapter presents the Phong reflection model. It is analyzed which parameters must be estimated to use the Phong reflection model to represent the captured radiances from a sample point. It is described how the parameters can be found using a non-linear least squares method. Finally, it is shown how the radiance distribution can be reconstructed from the parameters before results are shown. The theory behind the Phong reflection model is mainly based on [Ang06].

11.1 Phong Reflection Model

In computer graphics, one of the most used shading models is the Phong reflection model. The Phong reflection model is usually comprised of three terms; ambient, diffuse and specular reflection light. The contributions from the three terms to the complete model are illustrated in Figure 11.1 and explained in the following sections. It the following, all equations are for a single color channel, and therefore need to be evaluated for each of three channels.



(a) Ambient (b) Diffuse (c) Specular (d) Complete

Figure 11.1: The total illumination is the sum of the ambient, diffuse and specular term.

11.1.1 Ambient

The ambient term in the reflection model is used to approximate the global illumination caused by inter-reflection of light between objects in the scene. The ambient term is constant and is given by:

$$I_{\text{ambient}} = L_a \, k_a$$

In the equation, L_a is the global ambient light and k_a is the ambient color coefficient of the material. The ambient term is independent of the view direction and surface normal.

11.1.2 Diffuse

The diffuse term estimates the light which hits the surface and scatters in all directions. It is modeled as a Lambertian surface where light is scattered equally in all directions. The intensity is dependent on the angle between the surface normal and the incoming light. This is illustrated in Figure 11.2, where **n** is the normal vector to the surface and **l** is the vector pointing toward the light source. The more parallel the light vector is to the surface normal, the higher the intensity of the reflected light.

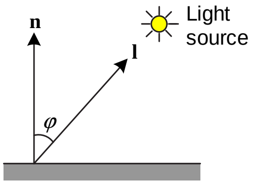


Figure 11.2: The diffuse term of the Phong reflection model.

The diffuse term is modeled using Lambert's law:

$$I_{\text{diffuse}} = L_i\, k_d\, \cos(\varphi) = L_i\, k_d\, (\mathbf{n} \bullet \mathbf{l})$$

In the equation, L_i is the incident light, k_d is the diffuse color coefficient of the material and θ is the angle between **n** and **l**. As with the ambient term, the diffuse term is independent of the view direction.

11.1.3 Specular

The specular term models the light that is reflected more in certain directions. On glossy surfaces, a highlight can be observed when looking at it from certain angles. The intensity of a highlight is dependent on where the surface is seen from. This is illustrated in Figure 11.3, where **v** is pointing towards the eye and **r** is the mirror direction of the light.

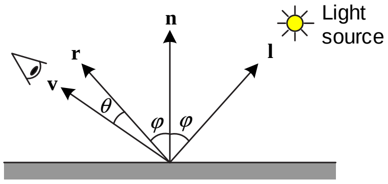


Figure 11.3: The specular term of the Phong reflection model.

The intensity of the highlight is dependent on the angle between the **v** and **r**. A strong highlight is seen when this angle is zero. The highlight decreases as the angle increases. The relation between view direction and the perceived intensity of highlight is given by:

$$I_{\text{specular}} = L_i\, k_s\, \cos^m(\theta) = L_i\, k_s\, (\mathbf{v} \bullet \mathbf{r})^m$$

In the equation, L_i is the incident light form the source, k_s is the specular color coefficient of the material, φ is the angle between $\mathbf{v}$ and $\mathbf{r}$, and m is the shininess of the material. When m is infinite, the surface is a perfect mirror, where light is only reflected in one direction.

11.1.4 Complete Model

The complete Phong illumination model is the sum of the ambient, diffuse and specular term. When many lights are present in the scene, the diffuse and specular term are evaluated for each light source. The complete model is therefore given by for each color channel:

$$\begin{aligned} I_{\text{complete}} &= I_{\text{ambient}} + I_{\text{diffuse}} + I_{\text{specular}} \\ &= L_a\, k_a + \sum_{j=1}^{M} L_{ij}\ \left(k_d\,(\mathbf{n} \bullet \mathbf{l}_j) + k_s\,(\mathbf{v} \bullet \mathbf{r}_j)^m\right) \end{aligned} \tag{11.1}$$

11.1.5 Modified Phong Model

When using the Phong reflection model in a computer graphics application, the coefficients k_a, k_d, k_s and m is set for the material and L_i is set for each of the light sources. The light and view vectors are calculated from the light and viewer positions and the position of the fragment being shaded. If the model is to be used for modeling the radiance distribution from a sample point, the only known parameter is the view direction. The rest of the parameters must be estimated from the measured data. This poses a problem, since the model is able to use an arbitrary number of light sources. It must somehow be decided how many light sources to include in the model. This naturally dependends on the measured data. One approach is to manually set using knowledge about the lighting environment. Contrarily, a more adaptive approach can be taken where models using different numbers of lights are fitted to data to obtain the best fit. In this thesis, the number of lights is a priori known, and a more automatic and adaptive approach is not investigated.

Not all the unknown parameters in Equation 11.1 need to be estimated individually. Many of these can be collapsed into one single term, since the radiance distribution is only dependent on view direction. Therefore, Equation 11.1 can be rearranged into parts which are dependent and independent of the view direction for each color channel, as:

$$\begin{aligned} I &= \underline{i_a\, k_a + \sum_{j=1}^{M} L_{ij}\, k_d\ (\mathbf{n} \cdot \mathbf{l}_j)} \quad + \quad \sum_{j=1}^{M} \underline{L_{ij}\, k_s}\ (\mathbf{v} \cdot \underline{\mathbf{r}_j})^{\underline{m}} \\ &= \qquad\qquad K_d \qquad\qquad + \quad \sum_{j=1}^{M}\ K_{sj}\ \ (\mathbf{v} \cdot \mathbf{r}_j)^m \end{aligned} \tag{11.2}$$

Each of the underlined parts can be represented with a single term. K_d is completely independent of the view direction and can therefore be estimated as three parameters - one for each color channel. For each light source, two sets of parameters must be estimated: the product between incident light and specular term, K_s, and the mirror reflection direction, $\mathbf{r}$. K_s is one parameter per color channel. The mirror reflection direction is three parameters in cartesian coordinates (they could be stored by only two parameters in spherical coordinates). Lastly, the m parameter is the same for all light sources since it depends on the material.

In total, 10 parameters have to be estimated in the model, if only one light source is present. For each extra light source, six extra parameters must be estimated. This can be expressed as:

$$\underbrace{1}_{m} + \underbrace{3}_{K_d(r,g,b)} + (\underbrace{N}_{\#\text{lights}} \cdot (\underbrace{3}_{K_s(r,g,b)} + \underbrace{3}_{\mathbf{r}(x,y,z)})) \quad = \quad 4 + (N \cdot 6)$$

The sizes of parameter maps of different dimensions are listed in Table 11.1. The parameter maps contains SxT texels. It is very evident that the parameter map takes up significantly less space. For 100 view directions, the parameter map with 10 parameters consumes 50 times less space than the observation map. For 300 view directions, the factor saved is 150. If using 16 parameters (2 light sources), the factors are 30 and 90, respectively.

SxT	**32x32**	**64x64**	**128x128**	**256x256**	**512x512**
Observation map (N=100)	1.95	7.81	31.25	125	500
Phong parameter map (10)	0.04	0.16	0.63	2.5	10
Phong parameter map (16)	0.06	0.25	1	4	16

Table 11.1: The sizes in megabytes of an observation map with 100 view directions and a Phong parameter map with 10 and 16 parameters. It is quite evident that significant amounts of space are saved when parameterizing the surface point radiance distributions using the Phong model.

When the parameters are known, the radiance in a given viewing direction can be reconstructed using Equation 11.2. In the following section it is described how these parameters can be found.

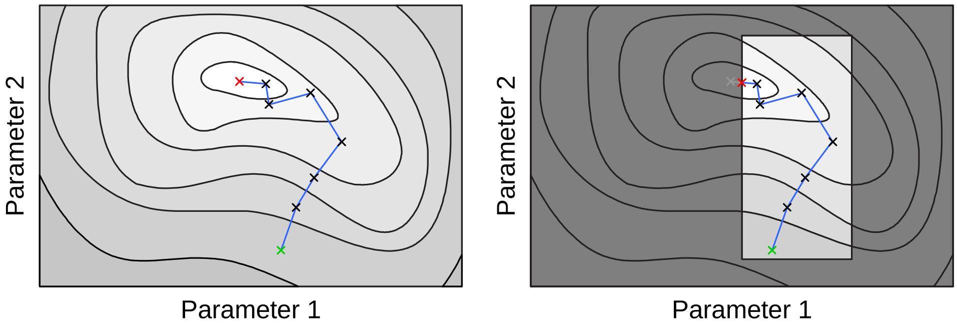
11.2 Estimation of Phong Parameters

The problem at hand is to estimate the parameters for the modified Phong model from measured radiances from a sample point. This is a non-linear curve fitting problem, where the parameters, $\bar{p}$, must be adjusted in such a way that the Phong model is as close as possible to the measured data. Normally, this is done by minimizing the sum of least squares:

$$\arg\min\left(\sum_{i=1}^{N}(\mathrm{Phong}(\mathrm{view}_i,\bar{p})-\mathrm{radiance}_i)^2\right) \tag{11.3}$$

The general approach to solve this problem is to use an iterative optimization method such as gradient descent or Gauss-Newton. These algorithms take a starting guess for the parameters and iteratively adjust these until a stop criterium is reached. The parameters are adjusted by examining the gradient at the point and taking a step in the steepest direction. The stop criterium can e.g. be a maximum iterations allowed or convergence of an error measurement, which means that a minimum is found.

Figure 11.4(a) illustrates a simple example, where two model parameters are optimized. The contours represent where the sum of residuals are equal. In the case of the Phong parameters the space is 10 dimensional, if one light source is used.



(a) Unconstrained

(b) Constrained

Figure 11.4: A fictive example of non-linear parameter estimation in 2D. The green cross is the initial guess, and the red cross is the optimal parameters.

The MATLAB `lsqcurvfit` function is used to estimate the parameters. It is based on the Reflective Newton algorithm [CL96]. It allows for setting constrains on the values that the parameters can obtain, as illustrated in Figure 11.4(b). Due to the constrains, another result is obtained compared to Figure 11.4(a). Both K_d and K_s are bounded between $[0,1]$ for each color channel, since the measured radiances are normalized to this range. The mirror reflection direction is limited to the upper hemisphere, and m is must be larger than one. Using these restrictions, a plausible solution is found, even though the global minimum might lie outside these restrictions.

In order to ensure that the algorithm converges and converges rapidly, a good initial guess for the parameters is needed. The diffuse component is estimated to be the mean of all the radiance samples. The mirror reflection is set to the direction from where the largest radiance is observed. Based on tests, it seems that the mirror reflection is the most important of the initial parameters to ensure convergence. The initial guess for the specular and shininess are not based on the observed data. The specular component is just set to 0 for each channel. This guess could be improved by looking at the maximum observed and the mean radiance. The shininess is set to 10. Using these estimates the algorithm converges in approximately 5-10 iterations for synthetically created radiance measures and 10-20 for real data.

The algorithm needs a minimum of four observations to work. Four observations give 12 equations since there are three color channels. At least 10 equations are needed since 10 parameters must be estimated. It is possible to get a model that looks correct using only four observations, if these are chosen appropriately. Both the diffuse and specular part must be seen. However, to ensure a good model fit of the radiance, many more observations are needed. The equation system is therefore overdetermined.

The following section tests how well the estimated parameters can model captured radiances.

11.3 Test

The test of how well-suited the Phong model is for modeling the measured radiances from a sampled point is divided into two parts. First, it is investigated how the model behaves with point light sources compared to area light sources. Secondly, the effect of assuming too few light sources in the model is tested.

The basic scene setup used in tests in this and forthcoming chapters looks as depicted in Figure 11.5. All tests are performed on synthetic data constructed in Autodesk 3ds Max using this setup. A square surface is created in the xy-plane. The planar surface has a size of 1000x1000, is centered at $[0, 0, 0]$ and has a normal of $[0, 0, 1]$. The surface is assigned different Phong materials depending on the test. In most of the tests, a single white point light source is used. The light source is placed at $[1500, 1500, 2000]$ and has a RGB color of $[1, 1, 1]$.

Modifications and additions to this basic setup are explained in the respective tests.

11.3.1 Point Light vs. Area Light

One of the issues with the Phong model is, that it assumes point light sources. This can by definition not be guaranteed in real scenes, where light sources all have an area. Therefore, this test investigates how the Phong model behaves when the surface is lit by a point light source compared to an area light source.

On the enclosed DVD, more figures from the test can be found in (◎`/Test/figures/test1/`). An application capable of reconstruction the surface from the test parameter map can be found in (◎`/Test/executable/`).

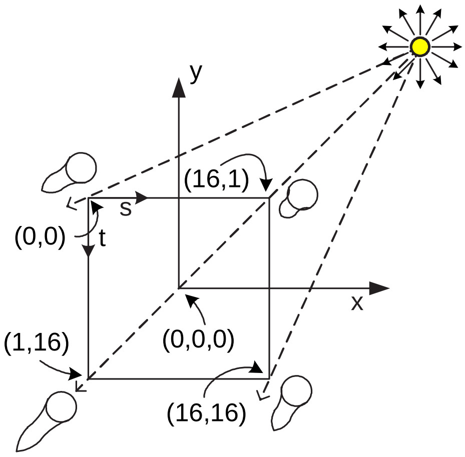


Figure 11.5: The basic setup in the following tests. One light source is placed in $[1500, 1500, 2000]$. The surface of size 1000x1000 is centered at $[0, 0, 0]$ and sampled in 16x16 points. The s, t coordinate system of the parameter map is shown as well as the corner points of it. Besides each of the surface corner points, the expected form of the fitted Phong model is sketched as seen from above.

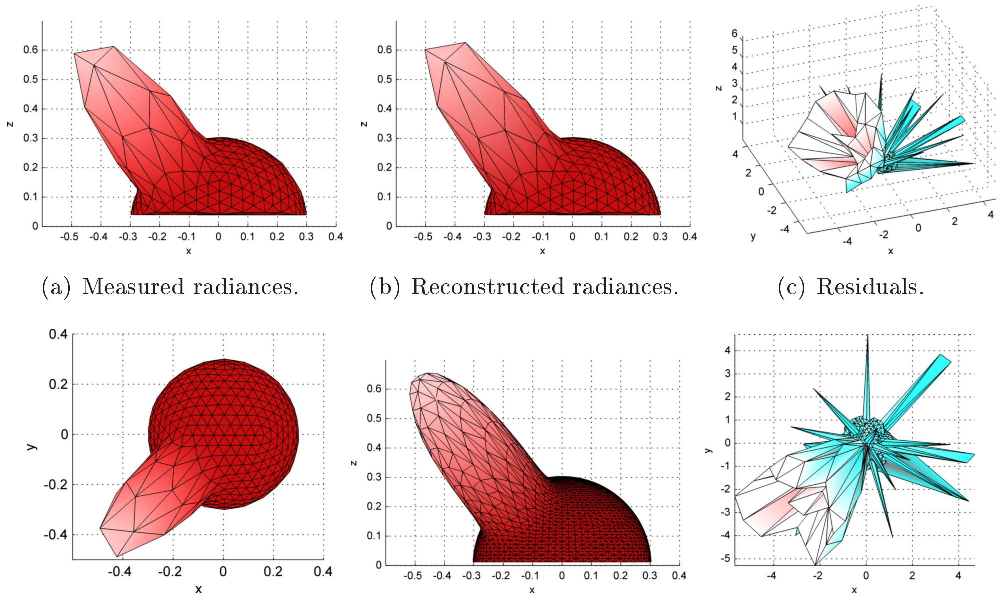
Results, Test Case 1: Point Light

The setup is as depicted in Figure 11.5. The color of the surface is $[200, 15, 15]$ in 8-bit RGB values. The surface is sampled from 289 directions following the icosahedron sampling scheme. The resulting models are expected to fit the data quite good as the setup fits the model well.

The results are visualized in Figure 11.6. Figures 11.6(a) and 11.6(d) show the measured radiances for a sample point seen from the side and from above. Figure 11.6(b) shows the reconstructed model. Figure 11.6(e) shows the reconstructed model plotted with a more dense grid to emphasize the smoothness. Figures 11.6(c) and 11.6(f) show the percentage residuals between measured and reconstructed radiances in all the sample directions.

The colors of the plots are given by the measured radiance, reconstructed radiance or residual values. This is why the residuals are predominantly cyan, as the mean residual errors are largest in the G and B color channels. It is clear that errors are more evident in the reflection direction. However, there are some spikes in the diffuse part. Still, the worst case errors are below 5% and the mean of the residuals are only 0.94%, 2.34% and 2.34% for the RGB color channels.

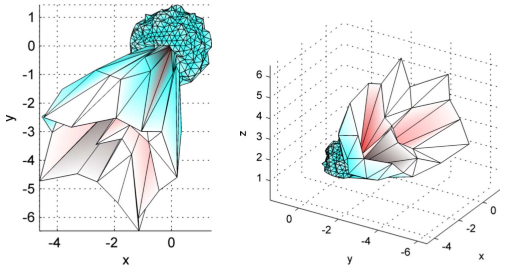
Figure 11.7 shows the residuals for the position $[14, 14]$ in the parameter map. These are included to indicate that, the even though residuals are much alike in both points, the random peaks in Figures 11.6(c) and 11.6(f) are not necessarily the general tendency. The spikes in the percentage residuals can be due to the very small values in the G and B color channels. Therefore, relatively small discrepancies in the estimation of parameters in these color channels can result in large percentage residuals.



(a) Measured radiances. (b) Reconstructed radiances. (c) Residuals.

(d) Measured radiances. (e) Reconstructed radiances. (f) Residuals.

Figure 11.6: The measured and reconstructed data for position $[8, 8]$ in the parameter map in test case 1 along with percentage residuals. The mean percentage residuals are $[0.94, 2.34, 2.34]$.



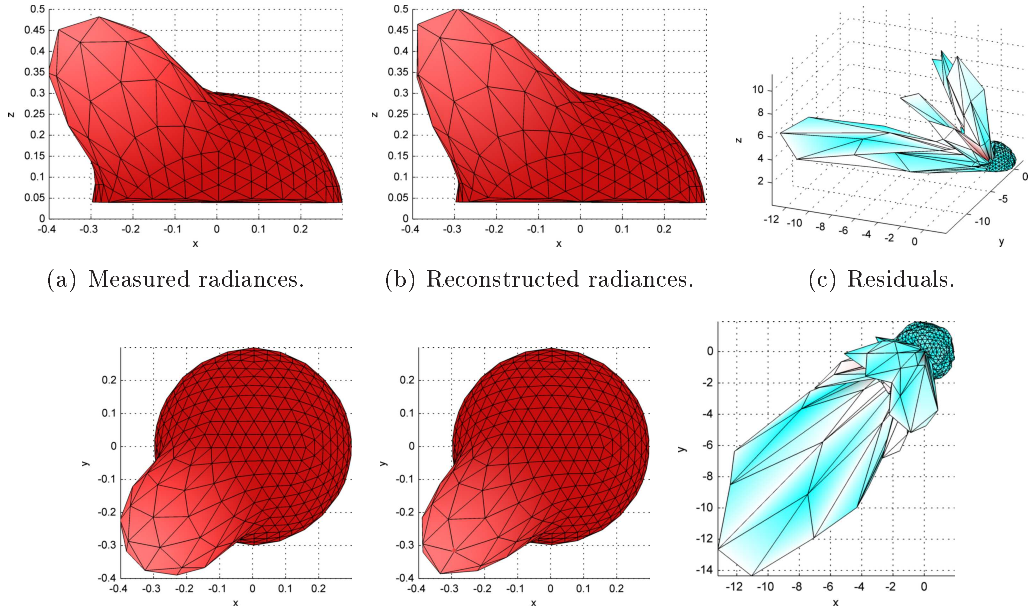
(a) Residuals (b) Residuals

Figure 11.7: The residuals for position $[14, 14]$ in the parameter map in test case 1. The mean percentage residuals are $[0.89, 2.27, 2.27]$.

Results, Test Case 2: Area Light

The setup is basically the same as in test case 1, but with the addition of 8 point light sources placed in the same height around the existing one. This is done to simulate an area light source. The color of the surface is still $[200, 15, 15]$ in 8-bit RGB values, and the surface is again sampled from 289 directions following the icosahedron sampling scheme. The sampled radiances are expected to show a more dampened and wide highlight than the one in test case 1. This is due to the fact that light does not only travel from one single direction to the point. It is not expected that the Phong model can fit the measured radiances as well as well as in test case 1. On the other hand, acceptable results are still expected.

The results are given in Figure 11.8. Figures 11.8(a) and 11.8(d) show the measured radiances seen from the side and from above. Figures 11.8(b) and 11.8(e) show the reconstructed radiances. The percentage residuals are shown in Figures 11.8(c) and 11.8(f). It is clear from the residuals that the model does not fit the radiances as well as in test case 1. The reconstructed radiances deviates from the measured data with approximately 10% at maximum. It is interesting to notice that the majority of errors lie “around” the highlight, indicating that the size of the highlight has been modeled sufficiently, but at the cost of a too narrow shape.



(a) Measured radiances. (b) Reconstructed radiances. (c) Residuals.

(d) Measured radiances. (e) Reconstructed radiances. (f) Residuals.

Figure 11.8: The measured and reconstructed radiances for position $[8, 8]$ in the parameter map in test case 2 along with percentage residuals. The mean percentage residuals are $[0.69, 4.30, 4.30]$.

The model fitting finds the shininess parameter to be lower than it actually is for the Phong material of the surface. This is because the light is an area light source, which results in a more wide highlight. This is the correct behavior, as the radiance distribution and not the materials are modeled.

Test Conclusion

All in all, the achieved results agrees with the expected results. The Phong model assumes point light sources, and therefore fits the point radiance distribution better for test case 1, where a point light source is used. However, for small area light sources the radiance distribution can be modeled sufficiently close.

11.3.2 Unexpected Number of Light Sources

Another issue with modeling the surface point radiance distribution using the Phong model is, that it is necessary to know how many light sources are present in the scene, i.e. that the model should incorporate. This test investigates how the radiance distribution is fitted with a one light source Phong model when multiple light sources are present in the scene. The light sources are placed further and further apart in three test cases. The basic setup is as in Figure 11.5, where one extra light source with a slightly lower intensity is added. In the three test cases the second light source is placed at $[450, 450, 2000]$, $[-450, -450, 2000]$ and $[-1500, -1500, 2000]$. It is expected that when the two highlights are close, the estimated Phong model might represent them as one. If they are further apart, it is expected that only the strongest of the highlights is modeled. This is expected because the initial guess on the highlight for the curve fitting process is based on the highest measured radiance.

On the enclosed DVD, more figures from this test can be found in (⊚`/Test/figures/test2/`). An application capable of reconstruction the surface from the test parameter map can be found in (⊚`/Test/executable/`).

Results

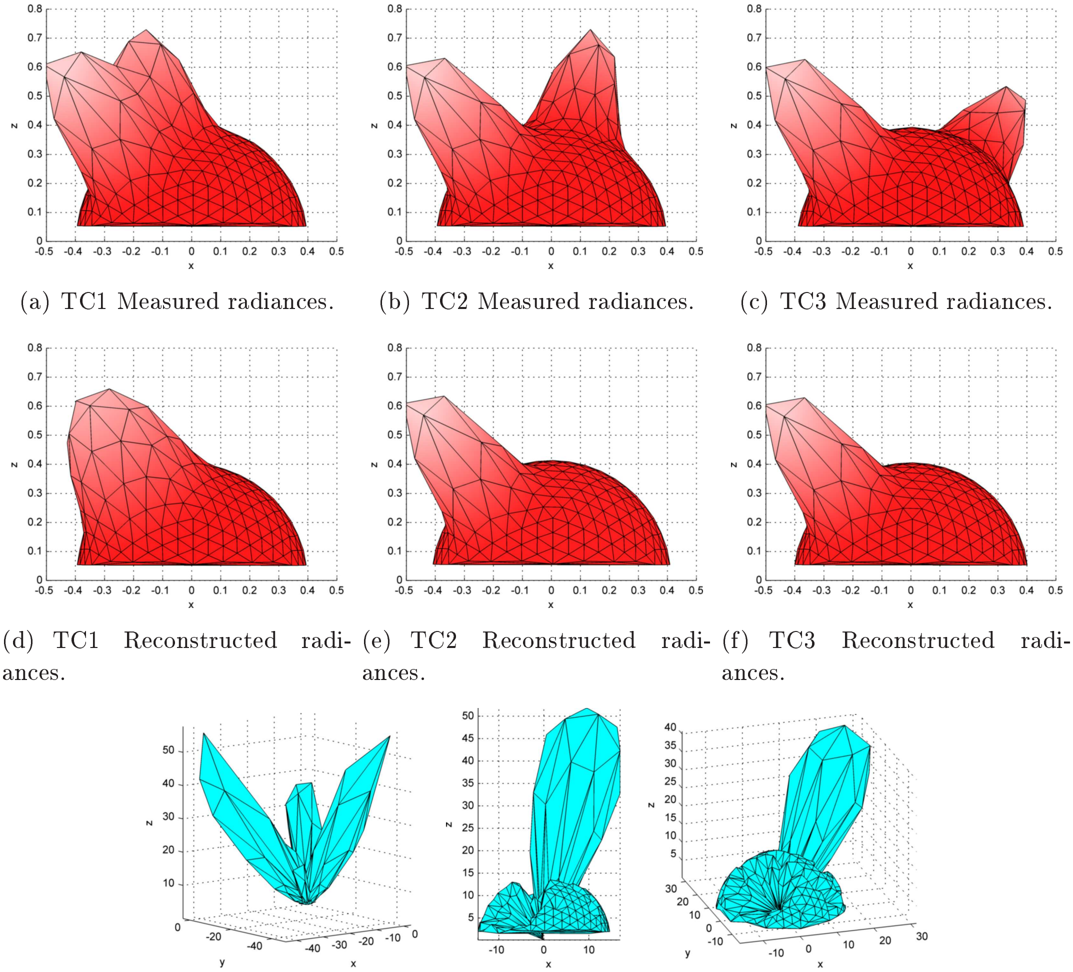
The results of all three test cases are summarized in Figure 11.9. The top row shows the measured radiances from all three test cases. The middle row plots the reconstructed radiances. The bottom row shows the percentage residuals for the figures in the corresponding column. In Figure 11.9(a) and 11.9(d), it can be seen that the two highlights are merged. The modeled highlight is wider than the two in the original data and its direction lies approximately in the middle of the two.

As the second light source is moved further away from the stationary light source, the highlights also move further away. This is evident in Figures 11.9(b) and 11.9(c). The reconstructed radiances are plotted in Figures 11.9(e) and 11.9(f). It is evident that only the strongest light source is captured in the model. The weak highlight is not modeled at all, which is also very visible in the percentage residual plots in Figures 11.9(h) and 11.9(i).

Test Conclusion

The Phong model assuming one light source is not able to capture multiple highlights in the measured radiance distribution. When the highlights are close, they are estimated as one. When they are further apart, only the strongest highlight is captured. This is that was expected.

An obviously area for further work is to investigate how correctly a Phong model with multiple light sources can estimate the data where more highlights are present.



(a) TC1 Measured radiances.

(b) TC2 Measured radiances.

(c) TC3 Measured radiances.

(d) TC1 Reconstructed radiances.

(e) TC2 Reconstructed radiances.

(f) TC3 Reconstructed radiances.

(g) TC1 Residuals.

(h) TC2 Residuals.

(i) TC3 Residuals.

Figure 11.9: The measured and reconstructed radiances for position [8, 8] in the parameter map in test case 2 along with percentage residuals. The mean percentage residuals are [0.00, 9.89, 9.89], [0.00, 20.58, 20.58] and [0.00, 22.07, 22.07].

11.4 Summary

This chapter has described how the radiances from a surface point can be modeled by a model inspired by the Phong reflection model. The model estimates the view-independent and view-independent parts of the radiance distribution separately. In order to use the model, the number of light source in the environment must be known. Furthermore, the light sources must be approximately point lights.

The test shows that when these restrictions are met, the radiances from a sample point can be accurately and compactly modeled using the Phong model.

12 Spherical Harmonics

This chapter introduces the theory behind spherical harmonics. It is described how the parameters of the model, that fits the radiance distribution the best, are found by projecting the captured radiances onto the spherical harmonics basis. It is also presented how the radiance distribution can be reconstructed from the parameters. The theory is mostly based on [Gre03]. The chapter is ended by showing results of fitting spherical harmonics to different radiance distributions.

12.1 Introduction to Spherical Harmonics

spherical harmonics (SH) are a linear set of basis functions, which can be used to represent spherical functions in a compact way. They are used in many different applications, ranging from computing atomic electron configuration to representing magnetic field of planets.

In computer graphics, SH is a fairly new technique. In [RH01], they use the SH basis for representing environment maps, which are used to illuminate diffuse objects. By using the SH, they can represent the environment with only 9 coefficients.

A method for rendering global illumination effects in real time is presented in [SKS02]. They use the SH basis functions for storing precomputed radiance transfer functions for each vertex in a scene. The transfer functions describe how incoming light at a point is transferred to light leaving the same point. The transfer functions are computed during a preprocessing step using a ray-tracer. By using the SH basis as a representation of both the transfer functions and the lighting environment, the integral part of the rendering equation[1] can be solved by the dot product of the SH-coefficients. The method is capable of rendering global illumination effects such as shadows and interreflection in real time for diffuse objects.

In this thesis, the SH basis functions are used to approximate the radiance measurements of surface points stored as texels in the observation map. After the following definition of the SH, it is described how it is used in this thesis.

12.2 Definition of Spherical Harmonics

The SH are an orthonormal basis. Basis functions can be thought of as small pieces of signal which can be scaled and combined to approximate a function [Gre03]. A well-known example of such a type of basis is the Fourier transform which decomposes a usually 1D

[1] See Appendix E for details on the rendering equation.

or 2D signal into its frequency components. SH are analogous to the Fourier transform, as they describe the frequencies of a signal defined over the surface of a sphere.

12.2.1 Associated Legendre Polynomials

At the heart of the SH are the associated Legendre polynomials, which are a set of orthonormal basis functions. The associated Legendre polynomials are usually denoted $P_l^m(x)$ and are defined over the range $[-1, 1]$. They are indexed by the two arguments, l and m, grouping the polynomials into bands. l only attains positive integers and defines the band index. m takes integer values in the interval $[0, l]$. The first three bands of the associate Legendre polynomials are plotted in Figure 12.1 with blue green and red, respectively.

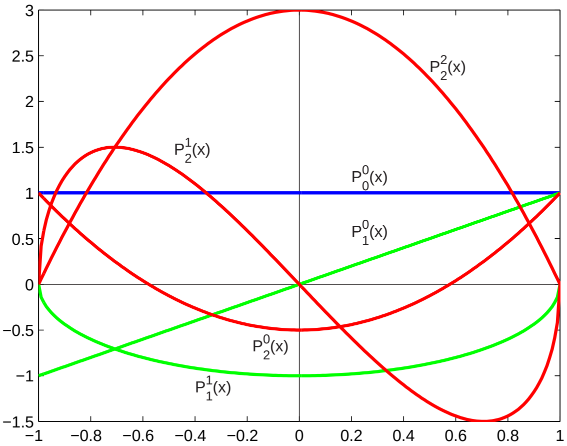


Figure 12.1: The first 6 associated Legendre polynomials.

The evaluation of the polynomials are usually done using the recursive definition in Equation 12.1. In order to evaluate e.g. $P_2^0(x)$, $P_0^0(x)$ is first calculated using rule 2. From this, $P_1^0(x)$ can be found from rule 3. Finally, $P_2^0(x)$ is obtained from rule 1. Generally, the highest $P_m^m(x)$ is found first, and rule 3 is used once to obtain $P_{m+1}^m(x)$. Next, rule 1 is used until $P_l^m(x)$ is found. For even integers, the double factorial is defined as the product of all smaller or equal, even integer. For odd integers, it is the product of all smaller or equal, odd integers.

$$\begin{array}{lrcl} \text{rule 1:} & (l-m)\ P_l^m & = & x\ (2l-1)\ P_{l-1}^m - (l+m)\ P_{l-2}^m \\ \text{rule 2:} & P_m^m & = & (-1)^m\ (2m-1)!!\ (1-x^2)^{m/2} \\ \text{rule 3:} & P_{m+1}^m & = & x\ (2m+1)\ P_m^m \end{array} \tag{12.1}$$

12.2.2 Spherical Harmonics Basis Functions

The SH basis function $y_l^m(\theta, \phi)$ is defined in Equation 12.2. The SH are evaluated in spherical coordinates[2].

$$y_l^m(\theta,\phi) = \begin{cases} \sqrt{2}\ K_l^m\ \cos(m\theta)\ P_l^m(\cos\phi), & m < 0 \\ \sqrt{2}\ K_l^m\ \sin(-m\theta)\ P_l^{-m}(\cos\phi), & m < 0 \\ K_l^0\ P_l^0(\cos\phi), & m = 0 \end{cases} \tag{12.2}$$

The l argument still denotes the band index and takes only positive integer values. However, m now takes integer values in the range $[-l, l]$. P is the associated Legendre polynomial. K is a normalization constant defined as:

$$K_l^m = \sqrt{\frac{(2l+1}{4\pi}\frac{(l-|m|)!}{(l+|m|)!}} \tag{12.3}$$

Often, it is more convenient to index the SH basis functions with i instead of l and m:

$$y_l^m(\theta,\phi) = y_i(\theta,\phi), \quad \text{where } i = l(l+1)+m \tag{12.4}$$

Reversely, l and m can be found from i as:

$$l = \left\lfloor \sqrt{i} \right\rfloor \qquad \text{and} \qquad m = i - (l \cdot (l+1)) \tag{12.5}$$

The first few bands of SH basis functions are shown in Figure 12.2. Each row represents a band. The columns are the m argument shown from $-l$ to l. It should be observed, that higher bands have higher frequency.

Approximating a function f describing the radiance distribution from a sample point by the SH basis functions is done by projection. The projection process determines how much f is like each of the basis functions. For each basis function, one coefficient is found. The projection is done by multiplying f with each basis function and integrating over the full domain of f, in this case the unit sphere where s denotes the direction:

$$c_l^m = \int_{\mathcal{H}_4} f(s)\ y_l^m(s)\ ds \tag{12.6}$$

When the coefficients are known, f can be reconstructed as $\widetilde{f}$ by summing all the basis functions weighted by the coefficients, where n is the number of used bands:

$$\widetilde{f}(s) = \sum_{l=0}^{n-1} \sum_{m=-l}^{l} c_l^m\ y_l^m(s) = \sum_{i=0}^{n^2} c_i\ y_i(s) \tag{12.7}$$

[2] See Appendix A for details on spherical coordinates.

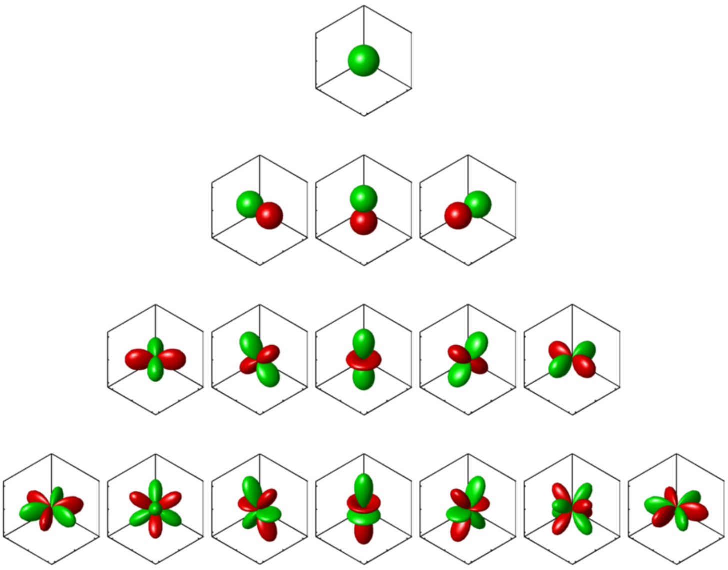
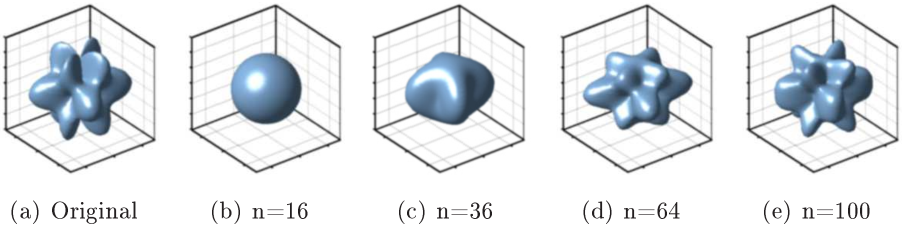


Figure 12.2: The first 16 spherical harmonic basis functions. Each row represent one band. Green represent positive values and red represent negative values. Figure based on [Gre03].

In order to reconstruct the correct function f, an infinite number of coefficients are in theory needed. This is obviously not feasible, and one must choose how many coefficients are needed to satisfyingly approximate the function. If the function is low-frequent, a few coefficients are typically sufficient. In Figure 12.3, increasing numbers of coefficients are used to reconstruct a complex function. In this example, approximately 100 coefficients are needed for a satisfactory reconstruction.



(a) Original (b) n=16 (c) n=36 (d) n=64 (e) n=100

Figure 12.3: The original function in A is reconstructed using different numbers of parameters.

In order to use the SH to represent the measured radiance distribution from a sampled point, each of the color channels are estimated separably. The size of the parameter map naturally varies with the employed number of parameters. Table 12.1 shows the sizes in megabytes of differently dimensioned parameter maps. It can be seen that when using 100 parameters pr channel, the SH parameter map takes up 3/5 the space compared to the observation. The fewer parameters used, the more space is saved in comparison to saving the entire observation map.

SxT	**32x32**	**64x64**	**128x128**	**256x256**	**512x512**
Observation map (N=100)	1.95	7.81	31.25	125	500
SH parameter map (16x3)	0.19	0.75	3	12	48
SH parameter map (36x3)	0.42	1.69	6.75	27	108
SH parameter map (64x3)	0.75	3	12	48	192
SH parameter map (100x3)	1.17	4.69	18.75	75	300

Table 12.1: The sizes in megabytes of an observation map with 100 view directions and an SH parameter map with 16, 36, 64 and 100 parameters pr color channel. It is evident that, the parameter map consumes less space than the observation map, even for many SH parameters.

12.3 Parameter Estimation Through Projection

In the previous section, it was stated that the SH coefficients can be found by projection. The projection involves integration over the surface of the unit sphere. Expressed in terms of θ, ϕ and differential solid angles, this becomes:

$$c_i = \int_{\mathcal{H}_4} f(s)\ y_i(s) ds = \int_0^{2\pi} \int_0^{\pi} f(s)\, y_i(s)\ \sin\phi\, d\phi\, d\theta \tag{12.8}$$

The question is how this integral can be solved. The following two subsections first present the usual approach and then the one employed in this work.

12.3.1 Monte Carlo Integration

When trying to solve the rendering equation, the integral over a sphere must be solved. A commonly used method for solving the integral is Monte Carlo Integration. The general idea in Monte Carlo integration is to create a large number of samples. The integral can then be solved by summing the sampled values each divided by their probability:

$$\int f(x)dx = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)} \tag{12.9}$$

This can be used to estimate the SH coefficients. If the samples are chosen uniformly with regard to area on the sphere, the probability for each sample is $1/4\pi$, which is moved outside the sum. The integral in 12.8 can be solved as:

$$c_i = \frac{4\,\pi}{N} \sum_{j=1}^{N} f(x_j)\, y_i(x_j) \tag{12.10}$$

Monte Carlo Integration works well, when it is possible to create a lot of uniform samples. This is possible in global illumination methods trying to solve the rendering equation. However, in this work it is not possible to guarantee that a texel in the observation map is sampled from directions spread equally on the hemisphere. Samples will often be scattered and some directions will be sampled more densely than others. Therefore, it is not possible to use the Monte Carlo integration method.

12.3.2 Numerical Integration

A slightly different approach to solving the integral is to use a basic numerical integration method. The integral of a 1D function can be estimated as a sum of rectangles, as illustrated in Figure 12.4. The area of each rectangle is the sample value multiplied with Δx. Normally, when using numerical integration, the samples are distributed uniformly. However, this is not a strict requirement, as it makes sense to sample more densely where the function changes rapidly. For the example in Figure 12.4 the integral can be approximated by:

$$\int_a^b f(x)\,dx \approx \sum_{i=1}^{N} f(x_i)\,\Delta x_i \tag{12.11}$$

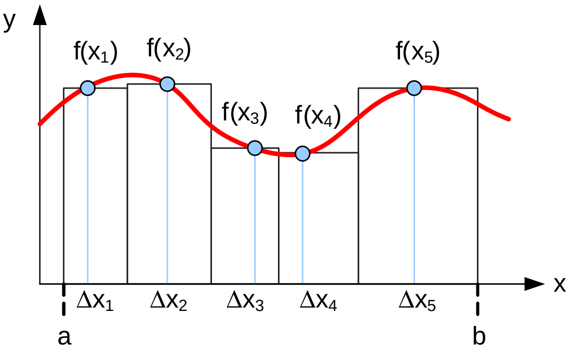


Figure 12.4: Example of numerical integration in 1D using non-uniform samples.

This technique can be extended easily to the integration of function defined over the surface of a sphere. The SH coefficients can therefore be estimated as:

$$c_i = \int_{\mathcal{H}_4} f(s)\,y_i(s)\,ds \approx \sum_{j=1}^{5} f(x_j)\;y_i(s_j)\,\Delta s_j \tag{12.12}$$

The coefficients are the sum of all sampled radiances multiplied with the SH basis functions and the area that the sample direction covers on the unit sphere. To use this method, the area Δs must be found for each sample direction. A method for this is presented in the following.

One way to partition a space is using Voronoi tessellation. Here, the space is divided into cells, which each contain all points closest to their corresponding sample. In order to use Voronoi tessellation on a sphere, the samples are projected onto a plane. This is described in [NLC02], where the samples on a unit sphere centered at $[0, 0, 1]$ are projected onto the plane $z = 2$ through a perspective transformation with origo as projection point. This is useful when partitioning on the entire sphere. However, since all radiance sample directions lie on the upper hemisphere, they can be projected onto the xy-plane by just omitting the z coordinate. In the xy-plane, the Voronoi diagram is constructed as illustrated in Figure 12.5.

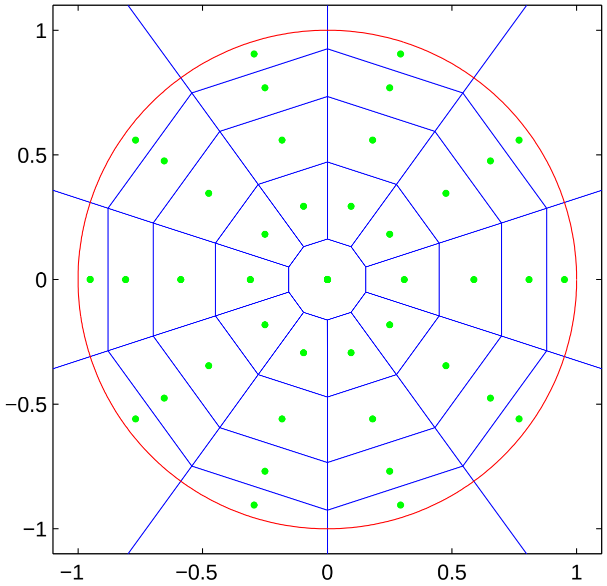


Figure 12.5: A voronoi diagram for samples in a latitude-longitude sample grid projected onto the xy-plane. The red circle is the unit circle which acts as a boundary.

The Voronoi Diagram is constructed by first triangulating the samples. Then, for each resulting triangle its circumscribing circle has its center at the corner of a Voronoi polygon. The polygons are constrained to the unit circle. The area of a Voronoi polygon, which corresponds to the area cover by a sample, can be found be splitting the polygon into simpler triangles and summing their areas. To calculate the area of Voronoi polygon on the sphere, it is weighted by $1/cos(\phi)$, where phi is the zenith angle of the sample.

Having found the area that a sample covers, it is possible to project the measured radiance distribution onto the SH basis using numerical integration.

12.3.3 Handling the Unsampled Hemisphere

The SH basis is defined over the entire sphere, but the radiances from a sample point are only sampled on the northern hemisphere. This naturally produces a problem, and there are different ways to handle this. In the paper about precomputed radiance transfer by [SKS02], integration is only performed over the northern hemisphere when projecting the data onto the basis. They call this approach zero hemisphere (ZH). [WAT92] use a reflection method, where the samples on the northern hemisphere are reflected onto the southern hemisphere. It is stated in [SHHS03] that the optimal way to handle the unsampled hemisphere is by projecting the ZH-coefficients onto a new basis using a "least-square optimal projection" (LSOP). [SHHS03] compares both the ZH, reflection and the

LSOP methods, and find that LSOP gives the best result.

In the LSOP method, it is sought to minimize the squared error, E, between the actual and the reconstructed samples to estimate the optimal coefficients vector $\mathbf{c}$:

$$E = \int_{\mathcal{H}_2} (f(s) - \sum_{i=0}^{n^2} c_i \, y_i(s))^2 ds \tag{12.13}$$

The function samples $f(s)$ and the SH basis functions are restricted to the northern hemisphere. Also the integral is over hemisphere and not the sphere. E is differentiated with respect to each of the coefficients c_k and set to zero to find the minimum. Therefore a number of equations equalling the number of coefficients are obtained:

$$\frac{\partial E}{\partial c_k} = \int_{\mathcal{H}_2} \Big(2 \cdot \Big(f(s) - \sum_{i=0}^{n^2} c_i \, y_i(s) \Big) \cdot y_k(s) \Big) ds = 0 \tag{12.14}$$
$$\Rightarrow \sum_{i=0}^{n^2} c_i \int_{\mathcal{H}_2} y_i(s) \, y_k(s) \, ds = \int_{\mathcal{H}_2} f(s) \, y_k(s) \, ds$$

This can be expressed as $\mathbf{A}\,\mathbf{c} = \mathbf{b}$ or $\mathbf{c} = \mathbf{A}^{-1}\,\mathbf{b}$, where $A_{ik} = \int_{\mathcal{H}_2} y_i(s) \, y_k(s) \, ds$ and $b_k = \int_{\mathcal{H}_2} f(s) \, y_k(s) \, ds$. $\mathbf{A}$ is a symmetric matrix which can be precomputed using e.g. Monte Carlo integration with a large number of uniform sample points. The $\mathbf{b}$ vector is the normal projection integrated over the hemisphere.

The test section concluding this chapter, shows experiments where both ZH and LSOP are used.

12.4 Reconstruction of Model From Parameters

Regardless of how the SH coefficients are obtained, the reconstruction of the hemispherical function $\widetilde{f}(s)$ is the same. To recreate the radiance of a surface point in a given direction, all coefficients are multiplied with their corresponding basis function evaluated in the given direction s are summed:

$$\widetilde{f}(s) = \sum_{i=0}^{n^2} c_i \; y_i(s) \tag{12.15}$$

Examples of recreated radiance distributions are shown in the following test section.

12.5 Test

To test the ability of SH to fit a measured radiance distribution from a sampled point, the following test is performed. As SH are a smooth set of basis functions, they are not able to model very high frequent data without use of many parameters and without possibly introducing some ringing effect. Therefore, this test investigates how well the SH model can be fitted to a data set using an increasing number of parameters. In this chapter, two methods for handling the unsampled hemisphere are presented. This test also has the purpose of comparing these two methods. The test is only performed on synthetic data. The data is obtained in a similar manner to the test of the Phong model described in Section 11.3 on page 62.

The purpose of the test is twofold. The two presented methods are compared and the effect of the number of used parameters are investigated.

12.5.1 Number of Parameters and ZH vs. LSOP

The setup for this test resembles the one in Figure 11.5, but with a few changes. The stationary light source is made a little more intense, and another more weak light source is inserted at $[-1500, -1500, 2000]$. This gives expected measured radiances as shown in Figure 12.6 with one strong and one weak highlight pointing in opposite directions.

The purpose of this test is to be able to compare the modeling of the sampled radiances when using a varying number of SH parameters, i.e. including an increasing number of SH bases in the model. It is expected that the model is fitted better as more parameters are used. However, some ringing effects are expected when using many parameters, and the model might begin to incorporate noise in its representation of the sampled data. It is in general expected that LSOP performs better than ZH, because of the reasons described in Section 12.3.3.

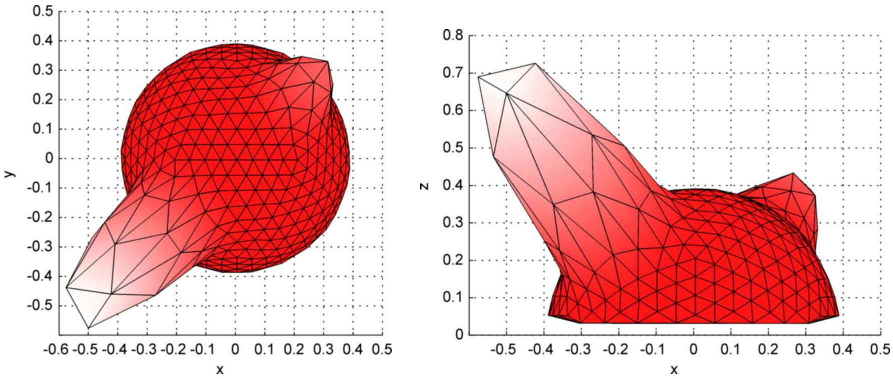
The test is comprised of six test cases, where the number of parameters are increased as follows: 16, 25, 49, 64, 81 and 100. Figure 12.6 shows the measured data, seen from above and the side. It is expected that fewer parameters are needed to model the measured radiances using LSOP than ZH.

On the enclosed DVD, more figures from the test can be found in (◎/Test/figures/test3/). An application capable of reconstruction the surface from the test parameter map can be found in (◎/Test/executable/).

Results

Figure 12.7 shows the results of fitting the SH model to the measured radiances with an increasing number of parameters. The method used for handling the unsampled hemisphere is ZH. It is quite apparent that the fitting is better as more parameters are used. This tendency is likewise visible in Figure 12.8, where LSOP is employed.

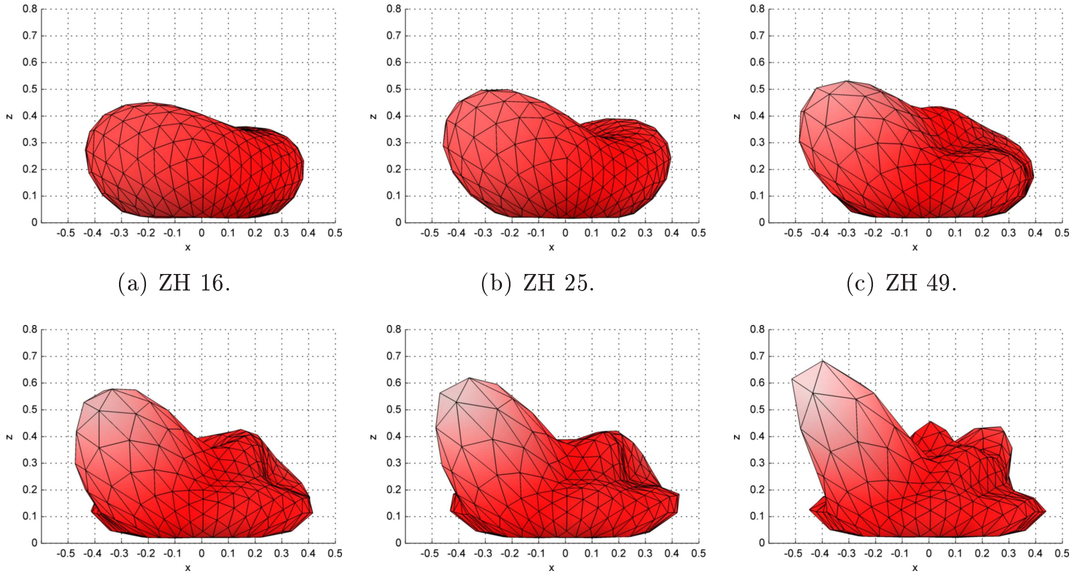
By inspection of the two image series, it is clear that a relatively high number of parameters are needed to sufficiently model the highlights in the measured radiances. With ZH, it is first when using 64 or more parameters that the smaller highlight becomes visible in the reconstruction. In all but the 100-parameter solution, the large highlight is consistently too wide and not strong enough. This issue is alleviated when using LSOP. The contour



(a) Measured radiances.

(b) Measured radiances.

Figure 12.6: The measured data used to investigate how well the SH model fits data as a function of the number of used parameters as well as differences in ZH- and LSOP approaches for handling the unsampled hemisphere.



(a) ZH 16.

(b) ZH 25.

(c) ZH 49.

(d) ZH 64.

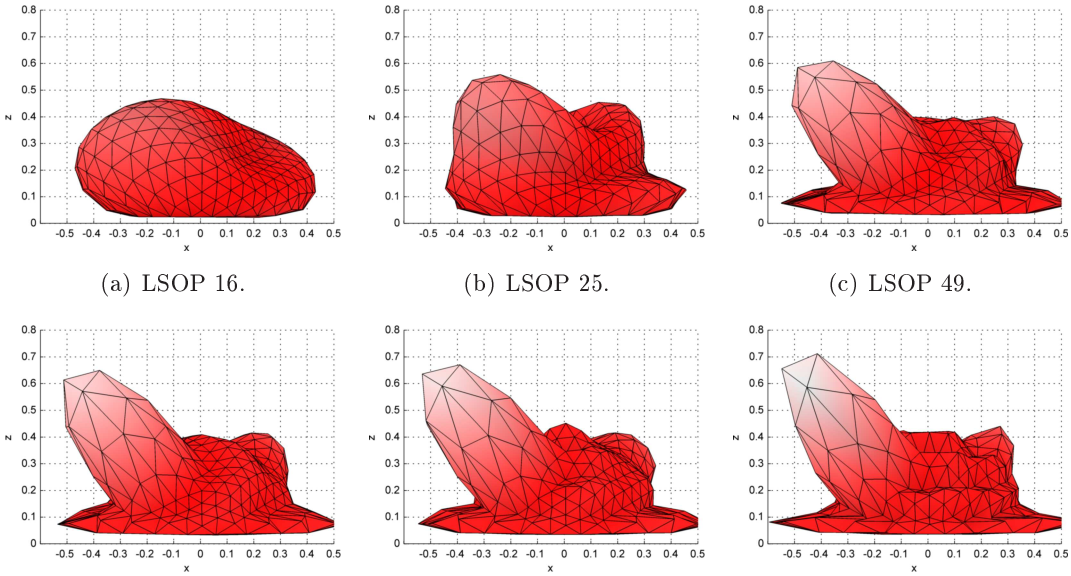
(e) ZH 81.

(f) ZH 100.

Figure 12.7: The reconstruction of the measured point radiance distribution in Figure 12.6 using SH ZH with increasing number of parameters.

of the smaller highlight is already visible with 25 parameters, and the large highlight is acceptably modeled with only 64 parameters. If looking at the lower row of Figure 12.8, it could be argued to not use more than 64 parameters, as more parameters seem to introduce ringing effects.

Even though the results in Figures 12.7 and 12.8 show similar overall tendencies, they do it with somewhat differing grace. The most apparent difference between the two series of images at first glance is the "disc-like abnormalities" that are present in the lower part of the LSOP-figures with 49 parameters and above. The reason for the abnormalities



(a) LSOP 16.

(b) LSOP 25.

(c) LSOP 49.

(d) LSOP 64.

(e) LSOP 81.

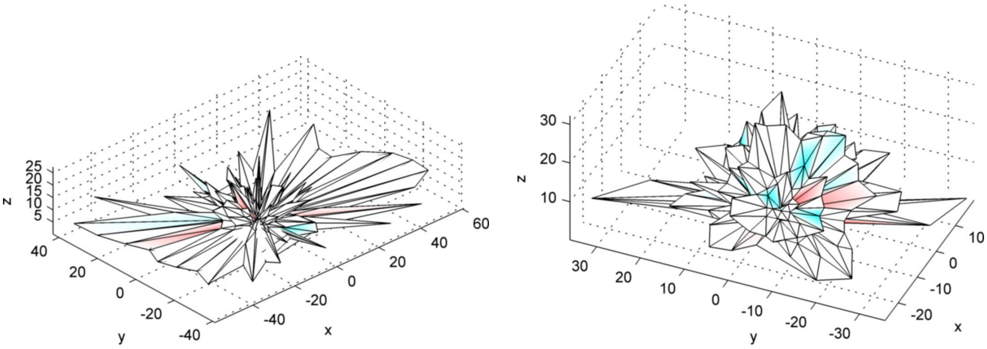
(f) LSOP 100.

Figure 12.8: The reconstruction of the measured point radiance distribution in Figure 12.6 using SH LSOP with increasing number of parameters..

could be the discontinuity close to the equator. It could also be an effect of incorrectly estimating the area that those samples cover when performing Voronoi tessellation. This phenomenon is clear when inspecting the percentage residuals plots for e.g. the 100-parameter LSOP result. A plot of this is shown in Figure 12.9(a), where residuals are present with magnitudes up to 40%. The corresponding residual plot for ZH is shown in Figure 12.9(b), where the residuals are almost as large. This is also clear when revisiting the plots of the reconstruction of the data. When comparing the very lowest part of Figures 12.6(b), 12.7(e) and 12.8(e) (those with high ϕ values), it is seen that neither of the two methods are capable of sufficiently modeling the measured radiance near the equator. The measured radiances have values of around 0.4 in this range, while the ZH reconstruction has values of around 0.25 and the LSOP reconstruction has values of around 0.55.

Another significant difference between the two methods that comes to attention when viewing the residual plots in Figure 12.9 is that the residuals for directions with $\phi < 75°$ are lower with LSOP than with ZH. This supports the statement that LSOP generally fits the measured data better. However, the mean residual values stated in the figure texts are actually almost identical. Without the errors around the high ϕ values, the LSOP approach would seem to have produced better results in this example.

One issue that is present with both approaches is ringing, which is a normal artifact when trying to model high frequent discontinuities in data with a combination of smooth basis functions. This is an effect of the model trying to fit the very high frequent change that occurs on the border between the northern and southern hemispheres because that southern hemisphere is unsampled. This could be solved with some kind of smoothing



(a) Residuals LSOP 100.

(b) Residuals ZH 100.

Figure 12.9: The percentage residual plots of the parameter maps with 100 parameters. The percentage residuals are $[11.61, 16.03, 16.03]$ for LSOP and $[7.67, 16.72, 16.72]$ for ZH.

function, which again might just tone down the highlights that LSOP is able to enhance. This idea is not investigated further in this thesis.

Test Conclusion

The expectation that LSOP is a better method than ZH is met. The statement in [SHHS03] that LSOP is a better method can thus be supported by data compiled in this thesis. This is mostly due to the fact that LSOP needs fewer parameters to model the highlights. With few parameters, ZH is unable to model anything useful. Only the diffuse part of the radiance distribution can be modeled. Some artifacts are detected with both methods, but their significance will be clear when using the models to reconstruct a sampled surface.

12.6 Summary

This chapter has presented how the radiance distribution for a texel in the observation map can be modeled using the SH basis functions. The parameters are estimated by projecting the radiance measurements onto the basis functions, where the measurements are weighted by their area on the unit sphere. Voronoi tessellation are used to estimate their areas.

The test shows that LSOP are generally better than ZH to represent the highlights using only few parameters. Ringing is introduced in the reconstructed radiances, because of the discontinuity at the equator. This is an issue which is not handled.

13 Comparison Test

This chapter investigates the abilities of Phong and SH LSOP to model the radiances from a sample point in different scenarios. It is tested how shiny a surface surfaces that can be modeled with the models. Next, the effect of the density of the sample directions is tested. Finally, a non-uniform sampling scheme where the highlight is sampled more densely is tested tested against a uniform sampling scheme.

In the previous two chapters, the Phong model and SH basis are presented. Each has their own advantages and disadvantages. The Phong model uses few parameters, but requires knowledge about the lights sources. The SH LSOP can handle more general lighting environments, but can not model high radiances from very specular surfaces without using lots of parameters. In this chapter the two methods are compared to investigate how they perform in three different scenarios which are presented in the following three sections. Only SH LSOP is used, because it proved to be better than SH ZH in the previous chapter.

13.1 Increasing Shininess

The purpose of this test is to find out how shiny surfaces the two models can model correctly. It is expected that the Phong model can model a strong highlight well, as long as the lights are point light sources. The SH LSOP is only expected to be able to model softer highlights. This is tested by trying to fit both models to radiance distributions from increasingly shiny surfaces.

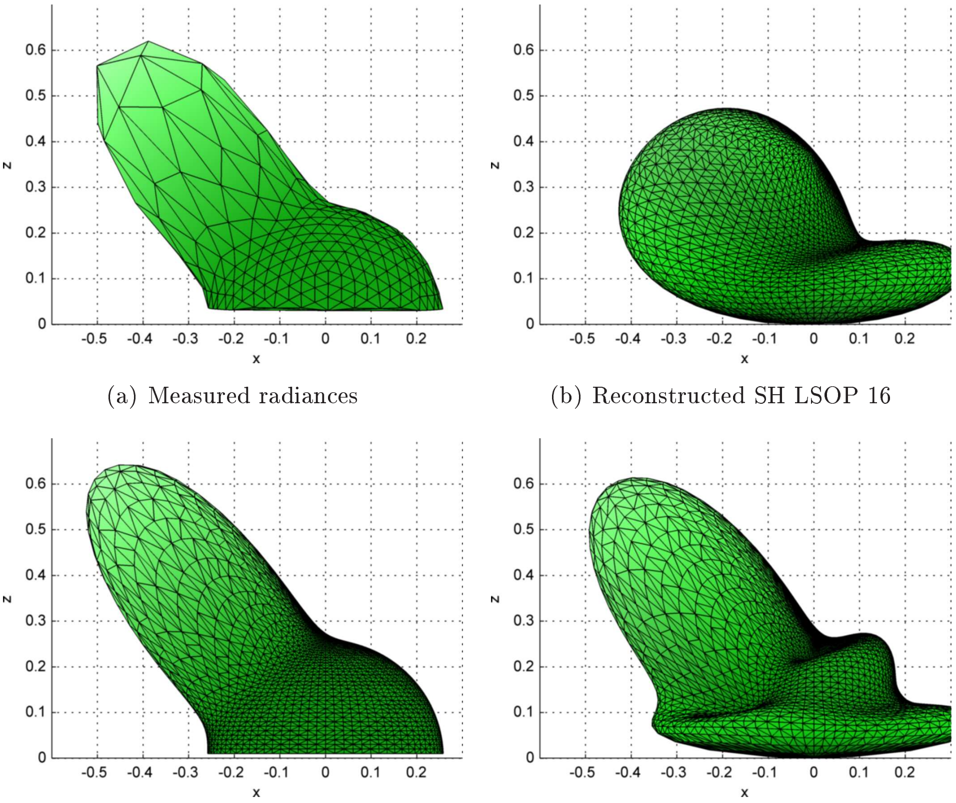
The basic test setup is the same as the one used in the preceding tests, described in Section 11.3 on page 62. Synthetic images of a surface lit by a point light source are rendered in 3ds Max. In total, 305 sample directions in an icosahedron grid are used. The surface is divided into eight bands with increasing shininess, ranging from very soft to very specular. All other properties for the bands are the same. The surface is sampled at 8x8 points so each band is modeled. For SH LSOP both 16 and 36 parameters are used to model the radiances.

On the enclosed DVD, more figures from this test can be found in (◎/`Test/figures/test5/`). An application capable of reconstruction the surface from the test parameter map can be found in (◎/`Test/executable/`).

13.1.1 Results

The results of the test is shown in Figure 13.1. Only the most specular band that the SH LSOP can model sufficiently is shown. All the more specular bands are not shown, because the SH LSOP is not able to represent these. The Phong model is able to model all the bands.

The measured data is shown in Figure 13.1(a). The highest band that the SH LSOP can model is the fourth band, which has a estimated Phong shininess coefficient of approximately 15. Figures 13.1(b) and 13.1(d) show the reconstructed radiance distributions using 16 and 36 parameters in the SH LSOP model. Using only 16 parameters the fit is not very good. Using 36 parameters the highlight is pretty well modeled, but the diffuse part suffers from ringing. For reference, the corresponding Phong model is shown in Figure 13.1(c).



(a) Measured radiances

(b) Reconstructed SH LSOP 16

(c) Reconstructed Phong

(d) Reconstructed SH LSOP 36

Figure 13.1: The result of the shininess test. The models are fitted to data from the fourth band. The estimated Phong model has a m factor of 15.

13.1.2 Test Conclusion

As expected, the Phong model can model both very soft and very strong highlights. The SH LSOP model needs many parameters and is still not able to capture strong highlights very well. Only a shininess of approximately 15 can be modeled using SH LSOP with 36 parameters. The conclusion is that in scenarios where the lighting environment have very few, small and well defined light sources, it seems better to model the surface with the Phong model than with the SH LSOP model.

13.2 Increasing Number of Sample Directions

The purpose of this test is to investigate the effect of the number of sample directions. The acquisition process gets easier, if few sample directions are sufficient to create an accurate model. For both models, it is expected that few sample directions are sufficient to model the sample point as long as parts of the highlights are sampled. More sample points should of course improve the models.

The test is performed using the usual setup (see Section 11.3 on page 62). The surface is sampled using three densities of sample directions. The sample directions are generated using the icosahedron scheme with different resolutions. The number of sample directions are 17, 73 and 305.

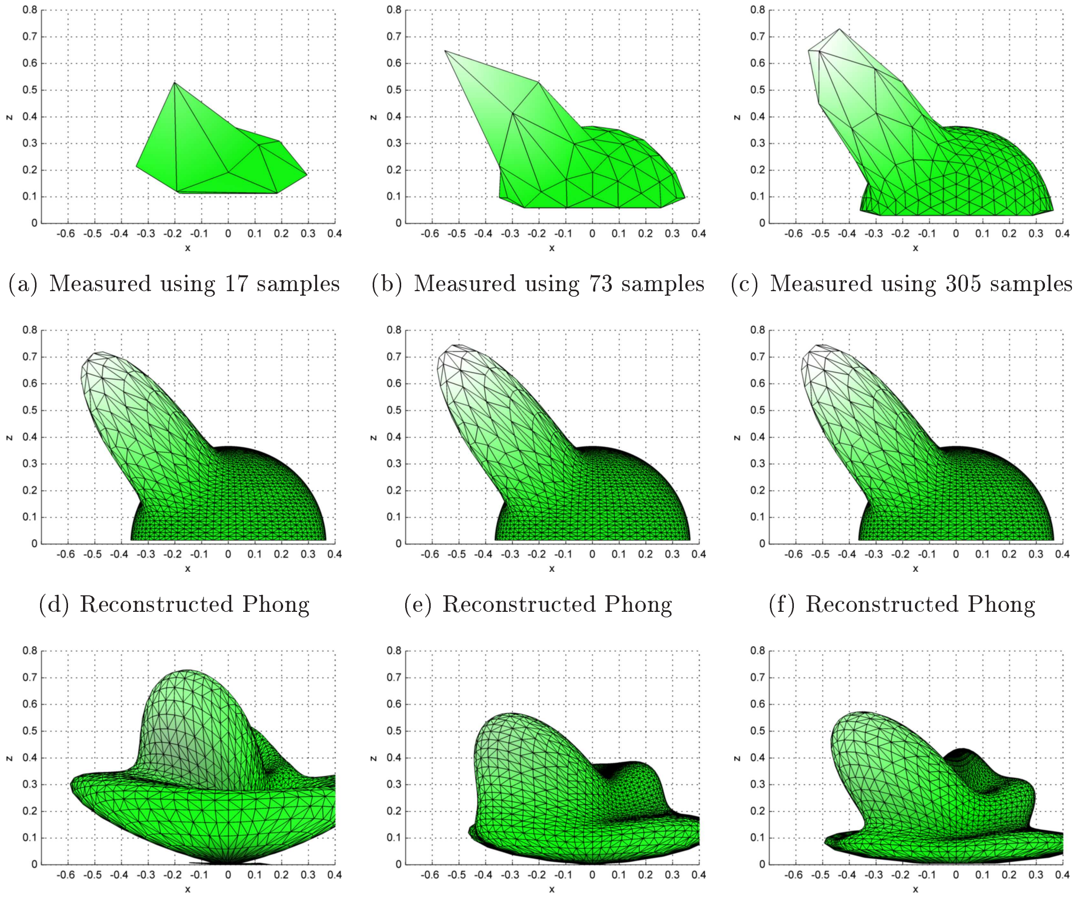
On the enclosed DVD, more figures from the test can be found in (◎`/Test/figures/test6/`). An application capable of reconstruction the surface from the test parameter map can be found in (◎`/Test/executable/`).

13.2.1 Results

The results of the three different sample direction resolutions are shown in Figure 13.2. The first row shows the three measured radiance distributions with different resolutions. The second row shows the reconstructed radiance distribution using Phong. The third row shows the reconstructed radiance distribution using SH LSOP with 36 parameters. Both Phong and SH LSOP are plotted using a high resolution sample grid and should be compared to the measured radiance in Figure 13.2(c).

It is clear that the difference between the Phong model estimated with many and few directions is relatively low. The difference between the estimated parameters for diffuse radiance and mirror direction when using 17 and 305 samples are below 1%. The largest discrepancy is for the specular parameters which differ with approximately 15%. The shininess parameter is about 5% off. So, even though the highlight is only sampled from very few directions, it is enough to estimate the Phong parameters within 15%.

For the SH, it is evident that 17 samples is not enough. For 73 samples the overall shape is captured, but the highlight is modeled to wide and not strong enough. Using 305 samples the shape of the highlight is almost captured, but it is to low. Using even more samples than what is shown here, no clear difference is obtained.



(a) Measured using 17 samples (b) Measured using 73 samples (c) Measured using 305 samples

(d) Reconstructed Phong (e) Reconstructed Phong (f) Reconstructed Phong

(g) Reconstructed SH LSOP 36 (h) Reconstructed SH LSOP 36 (i) Reconstructed SH LSOP 36

Figure 13.2: The effect of increasing the number of sample directions can be seen in the columns. The second row shows the estimated Phong models, and the third row shows the estimated SH LSOP models.

13.2.2 Test Conclusion

As expected, the Phong model parameters can be estimated from few sample directions. For the SH, more sample directions are needed to estimate the parameters. However, above a certain threshold, the effect of more samples are limited. Generally, no more than approximately 300 samples are needed, and for the Phong model less than 100 are sufficient.

13.3 Non-uniform Sampling

This test investigates the possibility for sampling a surface from directions spread non-uniformly on the hemisphere. When sampling a real surface from many directions, the acquisition is faster if only parts of the hemisphere needs to be sampled densely. Time could be saved by e.g. only sampling the highlight densely and sampling the remaining directions more sporadically. Instead of distributing a lot of a samples uniformly, fewer samples may be used, if they are from the important directions.

No reductions in the accuracy of estimated model are expected for neither the Phong or SH LSOP models even though fewer sample directions are used.

The test is performed using uniformly and non-uniformly distributed sample directions. The uniform sample directions are distributed using the icosahedron sample scheme resulting in 305 sample directions. The non-uniform samples are 73 uniformly distributed samples with the addition of 81 samples around the highlight, resulting in a total of 154 samples.

On the enclosed DVD, more figures from the test can be found in (◎`/Test/figures/test7/`). An application capable of reconstruction the surface from the test parameter map can be found in (◎`/Test/executable/`).

13.3.1 Results

The results are shown in Figure 13.3. The first row shows the uniform test case. The second row shows the non-uniform test. The first column is plots of the measured data. The highlight is better represented using the non-uniform scheme and fewer samples are "wasted" on the very even diffuse part.

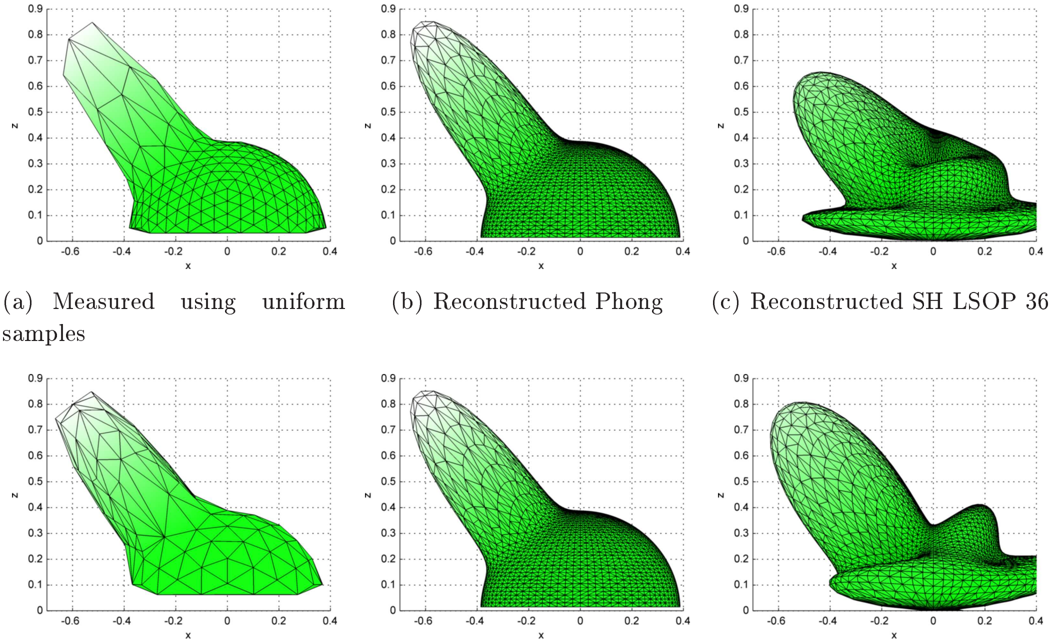
The second column plots the reconstructed data using Phong. No difference between the uniform and non-uniform sampling methods is apparent.

The third column shows the reconstructed data using SH LSOP with 36 parameters. It can be seen that the diffuse part is captured best using uniformly distributed samples. However the highlight is represented best with the non-uniform samples.

13.3.2 Test Conclusion

The parameters for the Phong model can be estimated accurately using few non-uniform distributed samples. No degradation can be seen between the Figures 13.3(b) and 13.3(e). Therefore, it is indeed possible to use a non-uniform scheme when estimating Phong parameters. It might even be possible with lot less samples.

No clear conclusion can be drawn for the SH. On one hand, the highlight is represented better with the non-uniform samples. On the other hand, the diffuse part is modeled worse. It seems to be a no-win situation, where one has to settle with one of the shown drawbacks.



(a) Measured using uniform samples

(b) Reconstructed Phong

(c) Reconstructed SH LSOP 36

(d) Measured using non-uniform samples

(e) Reconstructed Phong

(f) Reconstructed SH LSOP 36

Figure 13.3: The result of sampling the surface with an uniform and a non-uniform scheme. The first row shows the uniform test case and the second row shows the non-uniform.

13.4 Summary

Three different test scenarios have been investigated in this chapter. The first test shows that the Phong model can model both very soft and strong highlights. The SH LSOP can only model surfaces with a shininess up to approximately 15. If a strong highlight is present, more parameters than 36 seems to be needed.

The second test found that that no more than approximately 300 samples are needed to estimate the parameters of both the Phong and SH LSOP models. As few as 70 samples are sufficient for the Phong model, if the radiance distribution of the point does not have a very narrow highlight.

The last test found that a non-uniform sampling scheme works fine for estimating the parameters for the Phong model. However, mixed results were obtained for the SH LSOP model. The highlight is estimated better with the non-uniform samples, but the diffuse part is represented worse.

The choice of model to use depends on the complexity of the captured radiance distribution. If there are a known number of distinct highlights, it is best to use the Phong model. However for more diffuse and slowly varying radiance distributions the SH LSOP model seems to be a better choice.

Part IV
Visualization

This part accounts for the reconstruction of surfaces from parameter maps in 3D computer graphics applications.

The programmable processors of OpenGL are investigated in the introductory chapter of this part. This is done to examine how it is possible to implement a custom shader program on the graphics card. The goal is to perform more complex shading of fragments than what the fixed functionality pipeline in OpenGL provides, by using the parameter maps described in the previous part.

The second chapter details how shading of fragments are performed using the parameter maps that were constructed during model fitting. The flow of the custom shader programs are accounted for, and the overall pipeline for shading with parameter maps is shown.

The methods described in this part, acts as means for reconstructing the sampled surface in a view-dependent manner. They enable the comparison of a photograph of a real world surface with the reconstructed, virtual representation. Such comparisons are reserved for the next part concerning the evaluation of the system as a whole.

OpenGL Programmable Processors 14

The purpose of this chapter is to investigate the pipeline of OpenGL, how it presents itself with fixed functionality and how it is possible to replace certain parts of it to be able to recreate the surface from the parameter map. Vertex and fragment processing are detailed before the chapter is ended with a description of the execution model for custom shader programs in OpenGL.

Once the radiance from a surface has been sampled and models has been fitted to the obtained data, it is needed to be able to recreate the view-dependent appearance of the surface in a computer graphics application. In this thesis, this is done in OpenGL, by replacing the parts of the fixed functionality pipeline with custom shader programs.

14.1 OpenGL Fixed Functionality

Figure 14.1 shows the fixed functionality pipeline of OpenGL in a very condensed layout. The overall, basic functionality of OpenGL is as follows: take a number of vertices as input, perform geometric operations on them in the "Vertex Processing Machine", produce fragments which are candidates to pixels on the screen. Following this, the "Fragment Processing Machine" determines colors for fragments. In this step, pixel data can be used to e.g. paste a texture onto a primitive before showing it on the screen. Lastly, it is determined which of the fragments are to be put in the frame buffer (i.e. shown on screen).

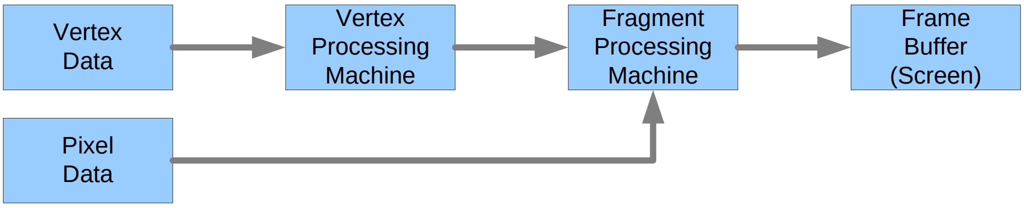


Figure 14.1: The fixed functionality pipeline of OpenGL. Figure after [SWND06].

As of OpenGL version 1.4, the main functionality of the two processing machines are replaceable by custom vertex and fragment shader programs. The following two sections go into more detail about the internals of the two processing machines and which parts of them can be replaced, before explaining the model for doing so.

14.2 Vertex Processing

As can be seen in Figure 14.2, the vertex processing pipeline operates on vertex position, normal, color/texture and fog information. By use of a number of transformations, this information is processed to prepare for primitive setup, clipping and the following fragment processing. It is important to notice that no knowledge of vertex connectivity is present or available in the vertex processor. Vertices are treated as singular entities, *before* primitive assembly.

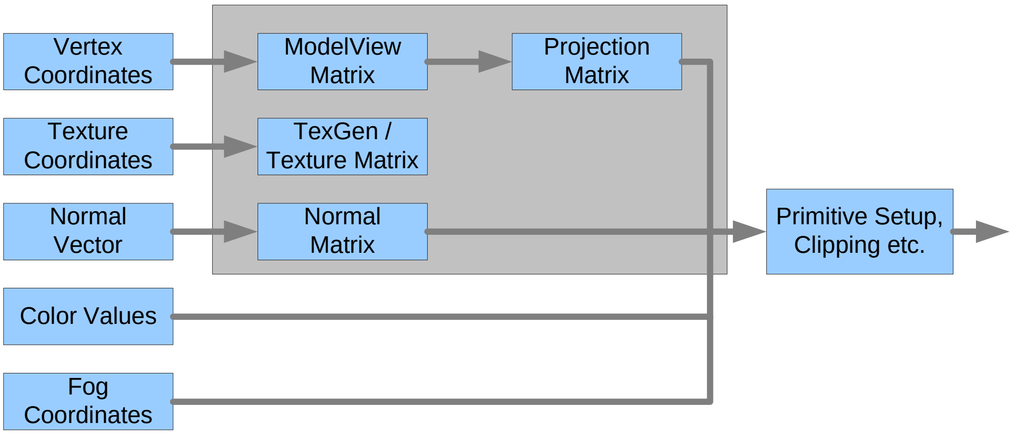


Figure 14.2: The vertex processing pipeline in OpenGL. The parts embodied in the gray box are replaceable by implementation of a custom vertex shader. Figure after [SWND06].

The gray box in Figure 14.2 encapsulates the steps in the fixed functionality pipeline, which can be customized using a custom vertex shader program. This is i.a. multiplication with the modelview and projection matrices on the vertex position, which can be modified to fit any desired behavior. The output of a vertex shader program must at least be the position of the current vertex. It is important to notice that a custom vertex shader can not replace just some of the fixed functionality, but must replace all stages in the gray box. However, if some stages are not needed, they do not need to be implemented in the vertex shader.

After the vertex processing machine, be it fixed functionality or not, has transformed vertex positions, a number of immutable operations are performed as the next step in the overall pipeline. This involves clipping, culling, rasterization and interpolation of vertex attributes for fragments.

14.3 Fragment Processing

The fragment processing machine receives a fragment along with ancillary interpolated vertex information such as normal, color and texture coordinates. It also has access to all texture units, and can therefore combine color and texture values to produce the final color for the fragment. This is illustrated in Figure 14.3, where the gray box encases the

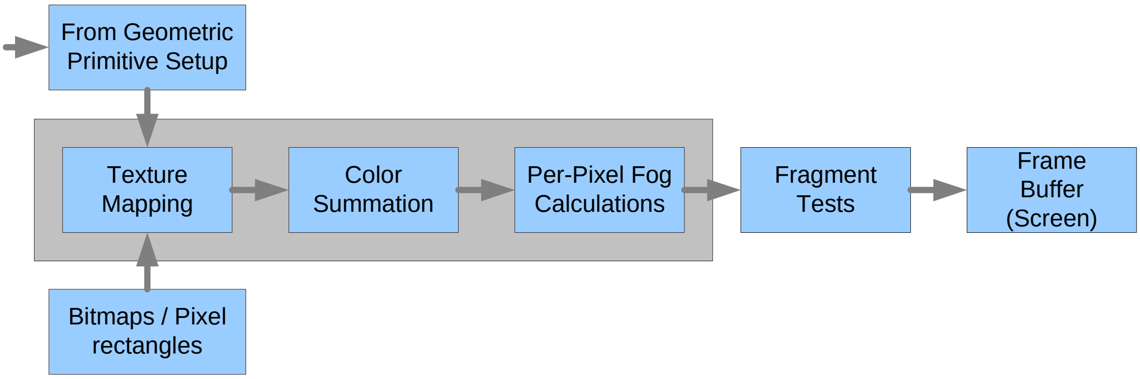

From Geometric Primitive Setup
Texture Mapping
Color Summation
Per-Pixel Fog Calculations
Fragment Tests
Frame Buffer (Screen)
Bitmaps / Pixel rectangles

Figure 14.3: The fragment processing pipeline in OpenGL. The parts embodied in the gray box are replaceable by implementation of a custom fragment shader. Figure after [SWND06].

stages customizable by implementation of a custom fragment shader program.

As indicated in Figure 14.3, the texture mapping, color summation and fog calculation stages are customizable in the fragment processing machine. This way, it is possible to arbitrarily combine any random part of any texture to produce a fragment color. The texture values can also be used for other purposes, such as dependent texture reading or as parameters to calculations. However, as it operates on single fragments only, it has no knowledge of neighboring fragments. Again, it is important to notice that a custom fragment shader can not replace just some of the fixed functionality but must replace all of the needed stages in the gray box.

As the last step in the overall pipeline, a number of tests and operations are performed on each fragment to determine whether or not it becomes a pixel on the screen. This includes pixel ownership, stencil, scissor and depth tests. It also includes alphablending. If the fragment passes all these tests, it becomes a pixel in the framebuffer or current render target.

14.4 Implementing a Shader Program

To implement a shader program in OpenGL, a number of steps must be undertaken. First, a shader object must be created. A separate shader object is created for each vertex and fragment shader. The application can then provide the source code for the desired shader, and have OpenGL compile it. This can be seen in Figure 14.4, where the upper layer represents the application, containing source code for the shader programs. The middle layer is the OpenGL driver, which is accessed through calls to the OpenGL API. The OpenGL driver then controls access to the graphics hardware in the lowest layer.

A program object must also be created. This holds the compiled shader objects, normally one for a vertex shader and one for a fragment shader. The program object can then be linked through OpenGL, which among other things checks for vertex/fragment shader

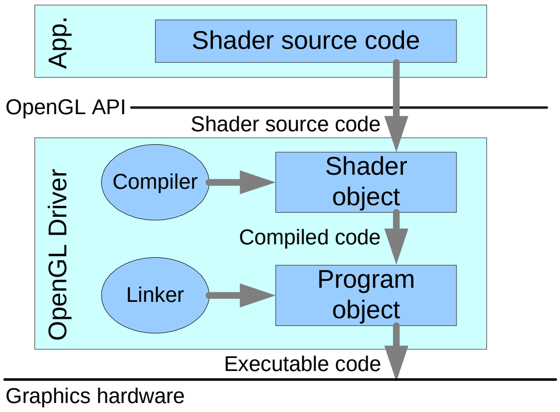


Figure 14.4: The execution model for OpenGL shader programs. Shader source code, defined in the application is passed to OpenGL, which performs compilation and linking, before passing it on to the graphics hardware for execution. Figure after [Ros06].

compatibility. The result is one or more executables (one for each shader to be customized) that can be run on the graphics hardware.

Any number of program objects can be created in the application, and the programmer is free to switch between them at runtime as necessary - but only one vertex and fragment program combination can be used at a time.

14.5 Summary

The purpose of this chapter was to find means for exploiting the power of modern graphics hardware. This enables possibilities for rendering surfaces using the parameter maps in real time. It is found that the fixed pipeline functionality of the GPU needs to be replaced to make it possible to use the parameter maps. By using custom shader programs, much more control of the rendering pipeline is gained. A custom shader program can be written that uses the values in textures as parameters to a model instead of as color value as usually done. How this is done is the topic of the following chapter.

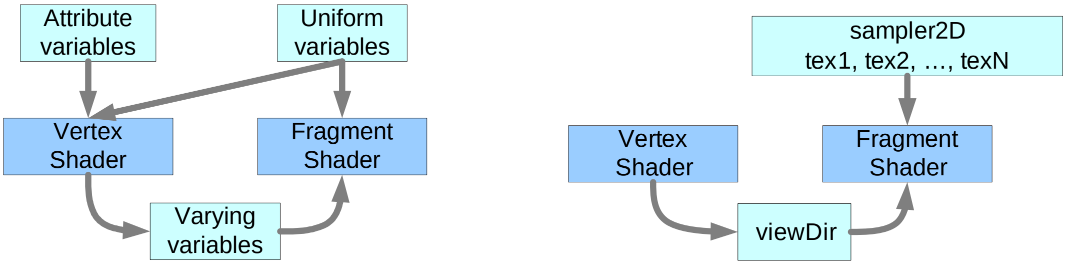
15 Parameter Map Shading

This chapter details how the actual shading using the parameter maps is performed. It is described how the parameters are communicated to the programmable shaders and how they are put to use there. Shading with both Phong and SH parameter maps is described.

In the previous chapter it was described how to replace the OpenGL fixed functionality vertex and fragment shaders. This is an important step in being able to shade surfaces using parameter maps with content as described in Chapters 11 and 12. There are two overall steps in using the parameter maps. It is necessary to get the values in the parameter maps communicated to the shader programs, and it is necessary to reconstruct fragment colors from a model using the parameter in the maps. The following sections detail how this is done for both models.

15.1 Shader Communication

OpenGL provides functionality to interface towards and between vertex and fragment shaders at runtime. Figure 15.1(a) outlines the model for this communication. The middle layer contains the two shaders and the two upper boxes show two types of variables whose values can be communicated from the application to the shaders. The bottom box shows a type of variable that can be communicated from the vertex shader to the fragment shader. The following listing explains these three types of variables and their use:



(a) Communication model for OpenGL shaders. Figure after [Ros06].

(b) Communication used in the parameter map shader.

Figure 15.1: Figures showing the general model for communication to OpenGL shaders and how it is used in the parameter map shader.

Attribute variables are used to pass frequently changing data into the vertex shader only. These variables can be changed once per vertex. The vertex shader can only read attribute variables and not write to them.

Uniform variables are intended for data that changes less frequently than attribute variables. They can be changed once per primitive. All vertex and fragment shaders contained within a shader program share a single global uniform variable name space, and two uniform variables with the same name in both the vertex and fragment shader are in fact the same variable. Uniform variables can only be read in the shader programs.

Varying variables are used to pass information from the vertex shader to the fragment shader, and is the only way to do so. Varying variables are written on a per-vertex basis and when the fragment shader reads the value, it gets an interpolated value - just as with color and normal values.

To be able to save all the parameters in a parameter map so they are accessible to the shaders, a number of textures are used. OpenGL provides functionality for pasting multiple textures to a surface. It is possible to combine the values in the textures in any desired way by implementing a custom fragment shader.

Instead of color values, it is possible to put arbitrary values into the textures and use them for other purposes in the shader, e.g. as parameters to a model for shading. It can then be decided whether to interpolate these parameter values over fragments or to use the nearest available value. Figure 15.1(b) shows the shader communication employed in the developed parameter shader. A number of textures are used to store the parameters of the model. These are enumerated using a corresponding number of uniform variable to indicate their texture unit indices. Lastly, as the appearance of the fragment should depend on the position of the viewer, an interpolated view direction vector is passed from the vertex to the fragment shader.

15.2 Implementing Parameter Map Shading

As the appearance of fragments using parameter maps depends on the view direction, it is needed to calculate a view direction on a per-fragment basis. Before presenting the actual shading with the two kinds of parameter maps, it is described how the view direction for each fragment is calculated.

The view direction vector is calculated in the vertex shader, and passed along interpolated to the fragment shader. A vector can be given in numerous different coordinate spaces, and one should therefore take care to ensure consistency within calculations. Figure 15.2 shows an overview of the coordinate spaces normally employed in OpenGL and how to move between them.

Normally, objects are defined within a local “object space” coordinate system placed somewhere which eases modeling. These are transformed using a model matrix to the “world space” coordinate system so objects can be placed in relation to each other. From here, a view matrix can be applied to transform the objects into “eye space” in which the origin is placed where the viewer is positioned. It is normal to concatenate these into a modelview matrix, which can move between the object and eye space directly. From the eye space, a projection matrix can be applied to go into the "clip space" where vertices that cannot be seen from the current viewing position can be excluded from the rendering for higher performance.

The leftmost box in Figure 15.2 represents the "tangent space". This is a space with a surface-local coordinate system, where the origin is located exactly in the point in question. The basis vectors for this space are a tangent to the surface at the point, the surface normal at the point and a bi-tangent which is the cross product of the other two vectors [Ros06]. The radiance distribution is modeled in a surface-local coordinate system for each sampled point. Therefore the view direction has to be transform into tangent space in the vertex shader and interpolated between vertex positions in the fragment shader.

In the following two sections is accounted for how the fragment shader programs for both the Phong and SH models work.

15.2.1 Phong Model Parameter Map Shading

As described in Section 11.1.5 on page 59, radiance using the Phong model can be reconstructed for one single light source as follows:

$$\begin{aligned} I &= \text{ambient term} + \text{diffuse term} + \text{specular term} \\ &= K_d + K_s\,(\mathbf{v} \bullet \mathbf{r})^m \end{aligned}$$

I is the radiance coming from the fragment in question. K_d is the combined ambient and diffuse term, K_s is the specular term, m is shininess and $\mathbf{v}$ and $\mathbf{r}$ are view and reflection directions. The combined ambient and diffuse term is saved as RGB color values as the first three parameters in the parameter map. The three next parameters are the specular RGB color values, followed by the 3 components of the reflection direction and lastly the shininess factor. The only thing needed to retrieve a view-dependent fragment color is the direction from which the fragment is viewed. The calculation of this vector is described in the start of this section.

With one light source and therefore 10 parameters, three textures are needed. A texture

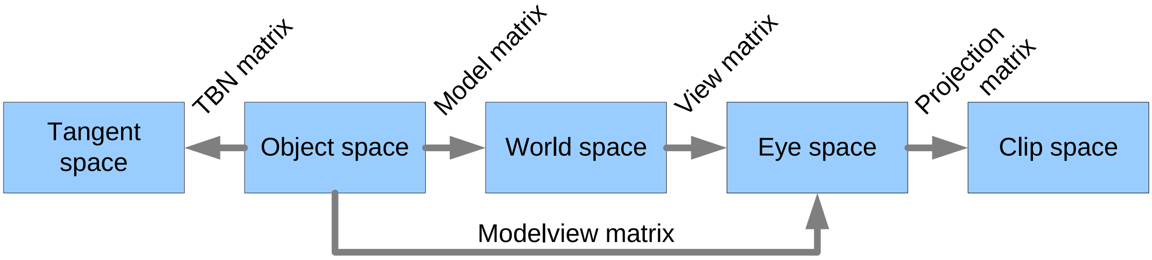


Figure 15.2: Different coordinate spaces used in OpenGL and the matrixes used to convert between them.

can contain up to 4 values per texel in RGBA format. In the current implementation, the K_d and K_s values are packed into two textures, and the m and $\mathbf{r}$ parameters are packed into the third texture.

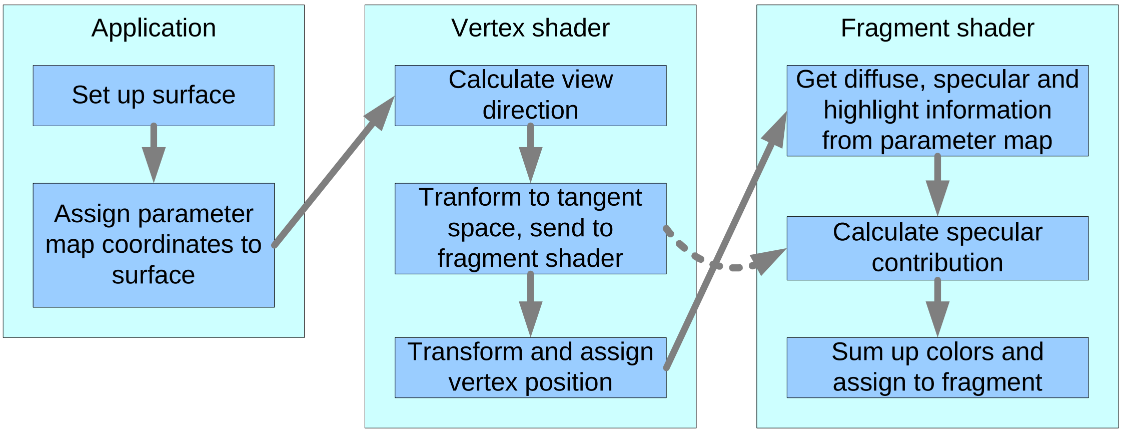

Application
Set up surface
Assign parameter map coordinates to surface
Vertex shader
Calculate view direction
Tranform to tangent space, send to fragment shader
Transform and assign vertex position
Fragment shader
Get diffuse, specular and highlight information from parameter map
Calculate specular contribution
Sum up colors and assign to fragment

Figure 15.3: An overview of the rendering pipeline for rendering using Phong parameter maps with one static light source.

The overall rendering pipeline for shading using Phong parameter maps is shown in Figure 15.3. This figure sums up the steps described in the above and is divided to show which processor takes care of which part of the process. The application, running on the CPU, sets up the surface and assigns parameter map coordinates to it, in the same manner as when texturing in OpenGL. Besides calculating the vertex position, the vertex shader also calculates the view direction vector in the tangent coordinate space and passes it to the fragment shader. Lastly, the fragment shader reads the parameters from the three textures and uses them in the corresponding model to assign a color to the fragment in question. The shader programs can be found on the DVD (⊚`/Application/executable/`) as `pm_phong.vert` and `pm_phong.frag`.

15.2.2 SH Model Parameter Map Shading

As stated in Equation 12.15 in Section 12.4 on page 75, a specific value of a reconstructed radiance distribution modeled with SH is found by multiplying all coefficients with their corresponding basis function evaluated in the given direction s, and summing up:

$$\widetilde{f}(\mathbf{v}) = \sum_{i=0}^{n^2} c_i \; y_i(\mathbf{v})$$

where c_i ranges over estimated coefficients, y_i is an SH basis function and $\mathbf{v}$ is the desired view direction. The view direction is calculated as described in the beginning of this section.

The number of parameters used for reconstruction is determined beforehand, and the number of textures and the structure of the custom shader are fitted to the number of parameters. The graphics card on the development computer has 16 texture units available[1]. Each texture can hold four parameters resulting in a total of 64 parameters. Since the a SH model is fitted for each color channel, there is only room for 20 parameters per channel, which in undesirably low.

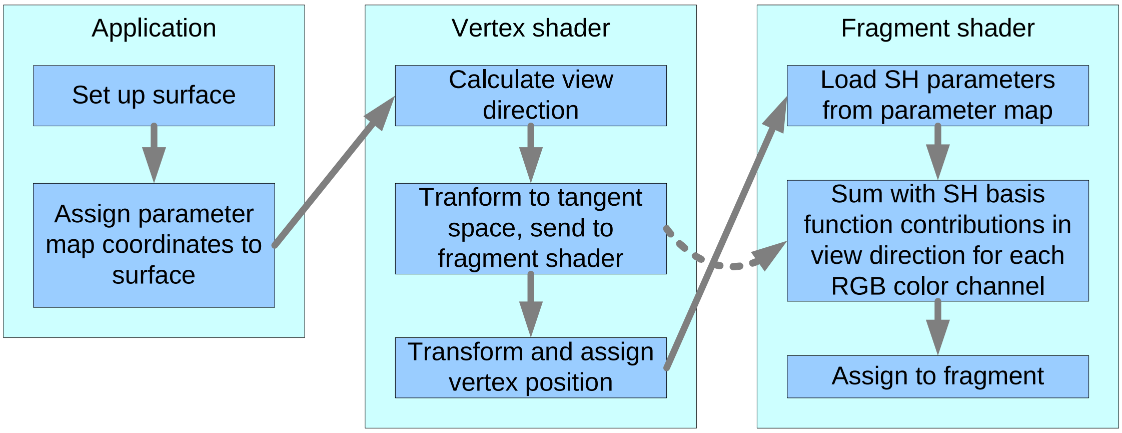

Application
Set up surface
Assign parameter map coordinates to surface
Vertex shader
Calculate view direction
Tranform to tangent space, send to fragment shader
Transform and assign vertex position
Fragment shader
Load SH parameters from parameter map
Sum with SH basis function contributions in view direction for each RGB color channel
Assign to fragment

Figure 15.4: An overview of the rendering pipeline for rendering using parameter maps, where the parameters are those to the SH model.

It is necessary to find other ways of packing more parameters into the available texture units. This could be done by e.g. employing techniques such as texture atlasing [nVI04] or by packing the parameters into multiple layers of a 3D texture. It is chosen to use texture atlasing and pack 16 textures into one. The drawback of this is, that the size of the parameter map is limited to 1024x1024 since the maximum texture size on the used GPU is 4096x4096. Using the atlasing technique up to 64 parameters can be saved in a single texture. The total number of needed textures is therefore three if 64 parameters are used per channel.

In the current implementation, it is possible to use either 16, 36 or 64 parameters in the SH model.

In a manner similar to that described in the section regarding Phong parameter map shading, Figure 15.4 sums up the steps involved with shading with SH parameter maps. Again, the main application sets up the surface and assigns parameter map coordinates to it. The vertex processor calculates vertex position as well as the view direction vector that is passed to the fragment shader. The fragment shader then sums up all contributions from the employed SH basis functions in each of the three color channels, and assigns this color to the fragment. The shader programs can be found on the DVD (◎`/Application/executable/`) as `pm_sh.vert`, `pm_sh16.frag`, `pm_sh36.frag` and `pm_sh64.frag`.

[1] The graphics card is a GeForce 7800 GTX on an AMD Athlon 64 3800+.

15.3 Summary

This chapter describes the communication between the developed shader programs and the application. When using either the Phong or SH parameter maps, the parameters are fed to the shader programs using textures. For both models, three textures are used to store the parameters. However, the SH parameter map is packed using texture atlasing.

For both types of parameter maps, the same vertex shader is used. The vertex shader transforms the view direction into a surface-local coordinate system in which the models are defined. Then the view direction is sent to the fragment shader. Two different fragment shader programs are developed for the two models. Using these shader programs, it is possible to render the captured surface at interactive frame rates. When using Phong parameter maps, 350 frames per second is achieved. Using SH parameter maps 125, 65 and 30 frames per seconds is achieved with 16, 36 and 64 parameters, respectively.

Part V
Evaluation

The purpose of this part is to evaluate upon the thesis in its entirety and present the results of the work.

The first chapter presents the results obtained with the methods described in the previous parts and discussion thereof. The results are presented as images of reconstructed surfaces seen from different view directions. The reconstructed images are compared to the original photographs from the same view direction to obtain a quantitative measure of the results. However, much of the results can also be obtained by qualitative comparisons of the two images and inspection of details. Discussion of the obtained results are also presented in this chapter.

The next chapter presents the conclusion upon the thesis as a whole. The contents of this thesis are summarized and general conclusions are drawn upon the work, where overall aspects are taken into consideration. The results of performed tests also provide a basis for conclusions upon the results of the work as well as being a basis for answering the initiating problem and problem statement

The part is ended with an outlook where ideas and topics for further work are presented and thoughts are given about what the future can bring for work within the field of capturing, modeling and recreating surface point radiance.

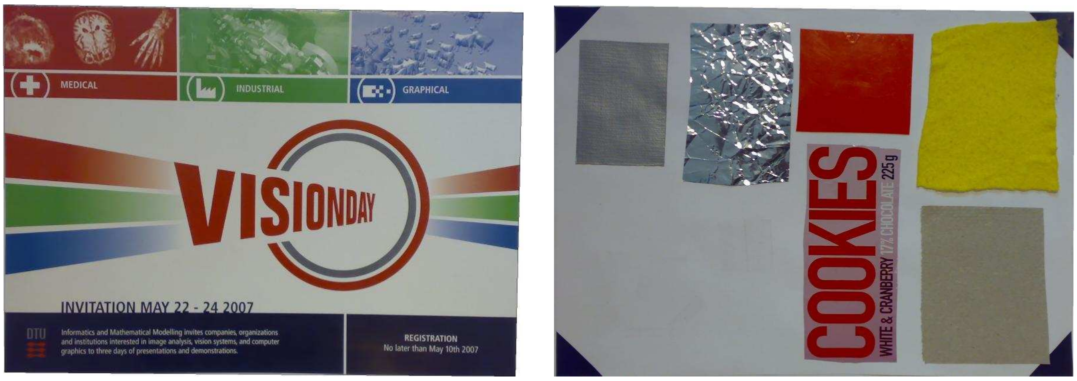
16 Results and Discussion

This chapter presents the test of the developed system as a whole. Two real world surfaces are photographed from numerous view directions, multiple points on their surface are sampled and models are fitted to their radiance distributions. The models are then used to reconstruct the surfaces. The renderings from the application are then compared to the photographs, both qualitatively and quantitatively.

16.1 Test Strategy and Setup

In the model fitting part, the Phong and SH models were only tested using synthetically generated data. This chapter tests the feasibility of sampling real surfaces and fitting the models to the sampled radiances for later reconstruction. To test this, a surface is sampled from a number of directions and the observation map is extracted. For all the texels in the observation map, both the Phong and SH models are fitted.

The test is performed on two different surfaces. Images of the two surfaces are given in Figure 16.1. The first surface is a coated paper flyer from the Visionday conference at DTU. It is more reflective than untreated paper, resulting in clear highlights from certain sample directions. The other surface is a patchwork of different materials ranging from very diffuse cloth to highly specular aluminium foil.


MEDICAL
INDUSTRIAL
GRAPHICAL
VISIONDAY
INVITATION MAY 22 - 24 2007
DTU
REGISTRATION
No later than May 10th 2007
COOKIES
WHITE & CRANBERRY 1/2 CHOCOLATE 225 g

(a) Visionday flyer

(b) Patchwork

Figure 16.1: Images of the two test surfaces. The first is a flyer for the Visionday conference at DTU. The second is a patchwork of different materials.

The tests are performed under controlled conditions. The surfaces are sampled in a dark room where the lighting can be controlled. To ease the acquisition process, a surface is

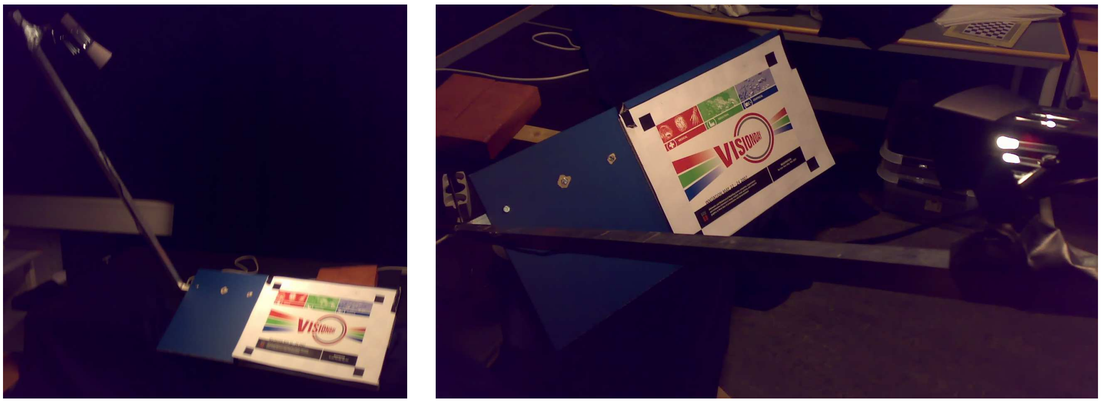


Figure 16.2: The setup used to acquire photographs of the surface from a lot of directions.

mounted on a platform, where it has to perpendicular rotational degrees of freedom, pan and tilt. Instead of moving the camera to vary the sample directions, the surface is moved. A small light source is also mounted on the platform, and it therefore maintains the same relative position to the surface. The light source is the only light present in the room. Photographs of the setup are depicted in Figure 16.2. In order to ease the registration of the surface, black squares are added at the corners of the surfaces.

After the samples have been acquired, images of a calibration target placed on the same rig are taken using the same camera settings. Using these images, the camera distortion can be handled in the construction of the observation map. The result of the camera calibrations can be seen in Figure F.2 in Appendix F on page 140.

After the observation map is created, both the Phong and SH models are fitted to the measured radiance for each texel. For SH, both the zero hemisphere and the least square optimal projection models are fitted. Both the reconstructed surfaces are applied parameter maps with a size of 512x512. Smaller maps can be seen in the interactive application on the DVD (◎/`Application/executable/`), where the size can be varied from 1x1 to 512x512.

The result of tests are judged both quantitatively and qualitatively by comparing the reconstructed surface to the appearance of the real surface. The real and reconstructed surfaces are compared quantitatively by looking at their difference image from different directions. The difference images are simply obtained by mapping the absolute difference between pixel values in the two images into pixel values in a third image. The qualitative test investigates whether the errors are noticable or the reconstructed surface appears correct.

The following two sections present the tests of the Visionday flyer and the patchwork. An executable application where it is possible to see the two surfaces and investigate the results is place on the enclosed DVD, (◎/`Application/executable/`).

16.2 Test: Visionday Flyer

The Visionday flyer depicted in Figure 16.1(a) is printed on reflective paper. It is mounted on the test platform seen in Figure 16.2. Videos and images of this surface can be found on the DVD (⊚`/Video/TestVideos_Visionday/`) and (⊚`/Test/figures/visionday/`).

16.2.1 Test Introduction

The Visionday flyer is a reflective piece of paper. The paper is almost completely flat. However, small bends and wrinkles are present. These bends and wrinkles cause the surface to reflect light unevenly. The printed parts of the flyer also seem to be more specular than the white parts.

The test is carried out using LDR photographs. The drawback of this is that the images taken where no reflection is present are very dark. The test is performed early in the work, to test the feasibility of the approach and developed tools, and the use of LDR images eased the acquisition process significantly. All images are taken using the same camera configuration.

The surface is sampled densely using 278 sample directions. The surface is sampled in a latitude-longitude fashion as described in Chapter 8. The zenith is kept fixed for a full rotation of the test platform. More sample directions are used for the larger zeniths than for the lower. This is done in an attempt to make the samples somewhat uniformly distributed over the hemisphere.

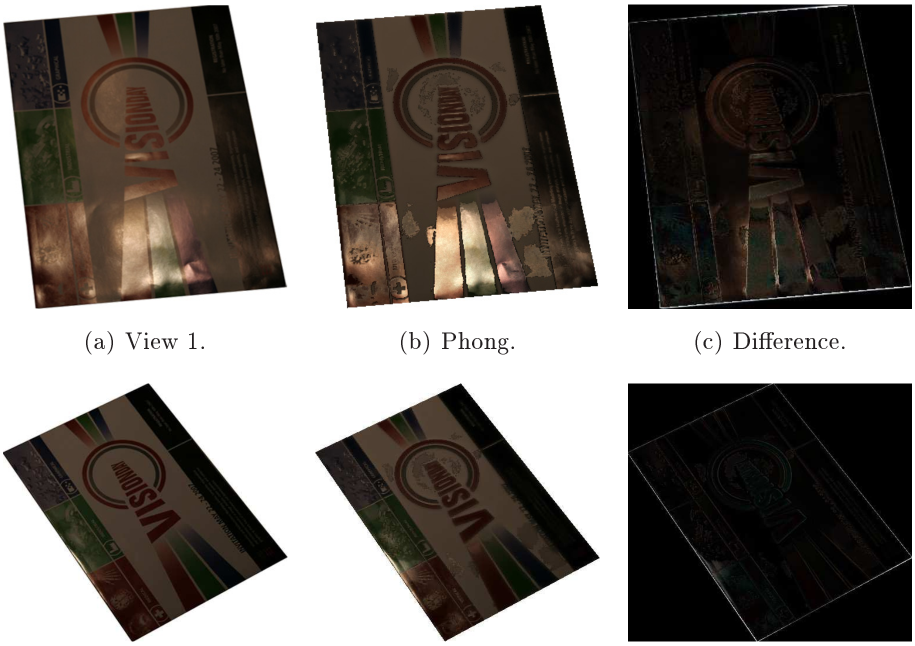
16.2.2 Test Results

Both Phong, SH ZH and SH LSOP parameter maps are created for the flyer. In the following, the results obtained for each of the models are presented.

Phong

The results for the Phong model are shown in Figure 16.3. Figures 16.3(a) and 16.3(d) show two real images of the flyer, taken from different angles. The first view has strong highlights in the bottom, while the second is overall more diffuse. The second column shows the reconstructed surface from the two views. The last column plots the difference images between the real and reconstructed images.

The outlines on the difference images are due to misalignment of the real and reconstructed images. The real images are slightly distorted which is an unavoidable error in the image acquisition. When rendering the reconstructed images, this distortion is not modeled. Also, the position of the camera is estimated through camera calibration which may be slightly off. In order to handle these discrepancies, the real and reconstructed images have been manually aligned.



(a) View 1. (b) Phong. (c) Difference.

(d) View 2. (e) Phong. (f) Difference.

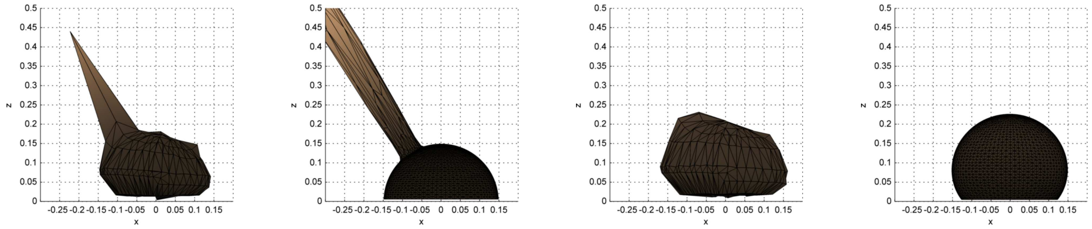
Figure 16.3: A comparison between the real and reconstructed flyer using the Phong model for two view directions.

The reconstructed and difference images in Figures 16.3(b) and 16.3(c), respectively, show that the highlight is captured well in the printed parts of the flyer. However, in the white parts of the flyer, the highlight is not captured everywhere. Stains appear in the white part which look different than the rest from all directions. Figures 16.3(e) and 16.3(f) show that the diffuse reflection distribution is captured very well.

Figure 16.4 shows the measured radiances for a sample point inside a stain and one outside stains, along with the estimated models. It can be seen, that the point inside the stain is modeled almost correctly. However, the highlight is too high because it is not sampled enough. Outside the stains, the highlight is not captured clearly from any of the sample directions. Also, the radiance is low for high zeniths.

It seems like the specular parameters of the Phong model have taken precedence when fitting the model to the diffuse data. Instead of being used to fit a specular highlight pointing out from the diffuse half-sphere, the specular parameters are used to model the diffuse half-sphere itself, as a vague specular highlight, perpendicular to the surface. Therefore, the reconstructed radiances are generally a bit too high, and the radiances at high zeniths go towards low values, because the specular part of the model has a “mushroom”-like appearance. This seems likely to be an effect of not being able to sample from very oblique directions with high zeniths and that recorded radiances at high zeniths are generally quite dark.

Another thing to notice in the renderings, is that bends and wrinkles are reconstructed



(a) Measured data from a sample point inside a stain.

(b) Reconstructed data from a sample point inside a stain.

(c) Measured data from a sample point outside the stains.

(d) Reconstructed data from a sample point outside the stains.

Figure 16.4: Plots of measured and reconstructed data from a sample point inside a stain, and a sample point outside the stains.

accurately as one would expect. This can be seen in the lower left corner and on the blue stripe in the bottom in the first view in Figure 16.3(b).

SH ZH

The results of the reconstruction using SH ZH are shown in Figure 16.5. Only the first view from the preceding test is shown. Figure 16.5(a) is the real photograph. The second column shows the reconstruction using 16 and 64 parameters. The third column shows the difference images.

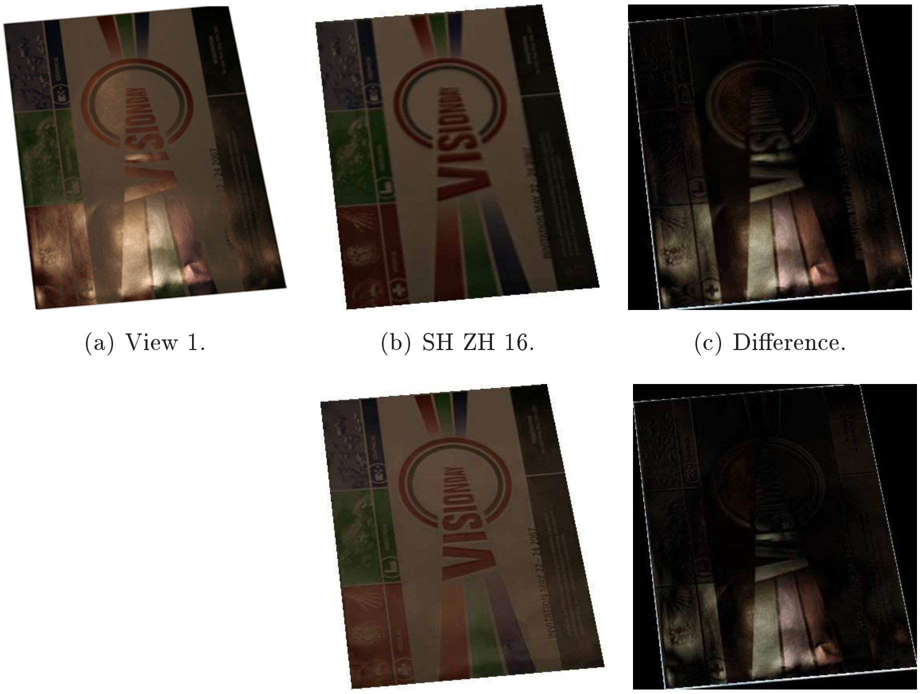
The renderings illustrate very well that strong highlights can not be modeled using SH ZH without the use of many parameters. This was also the finding in Chapter 12 on page 68. By using 64 parameters, the highlight is captured slightly better than when only using 16 parameters. This is most noticeable in the black part in the right edge the flyer. Also the bends in the blue stripe begins to appear when using 64 parameters. The diffuse part in the top of the flyer is modeled equally well with both 16 and 64 parameters.

SH LSOP

The results for using SH LSOP to model the radiance distribution are shown in Figure 16.6. The renderings are from the same direction as the preceding test. Figure 16.6(a) is the real image. The second column depicts the reconstructed images, and the third column shows the difference images.

When using 16 parameters, the result corresponds roughly to the results achieved using 16 parameters in SH ZH. It is still not possible to model the highlights. When looking at the SH LSOP 64 reconstruction, it is clear that something is wrong. In the white parts of the flyer there is a lot of noise present.

When plotting the radiances from a sample point from the white, noisy area in Figure 16.6(d), as done in Figure 16.7, it can be seen the errors are due to severe ringing in the



(a) View 1.

(b) SH ZH 16.

(c) Difference.

(d) SH ZH 64.

(e) Difference.

Figure 16.5: A comparison between the real and reconstructed flyer from the first view direction using 16 and 64 parameters in the SH ZH model.

reconstructed data. This was also seen when testing with synthetic data in Chapter 12 on page 68.

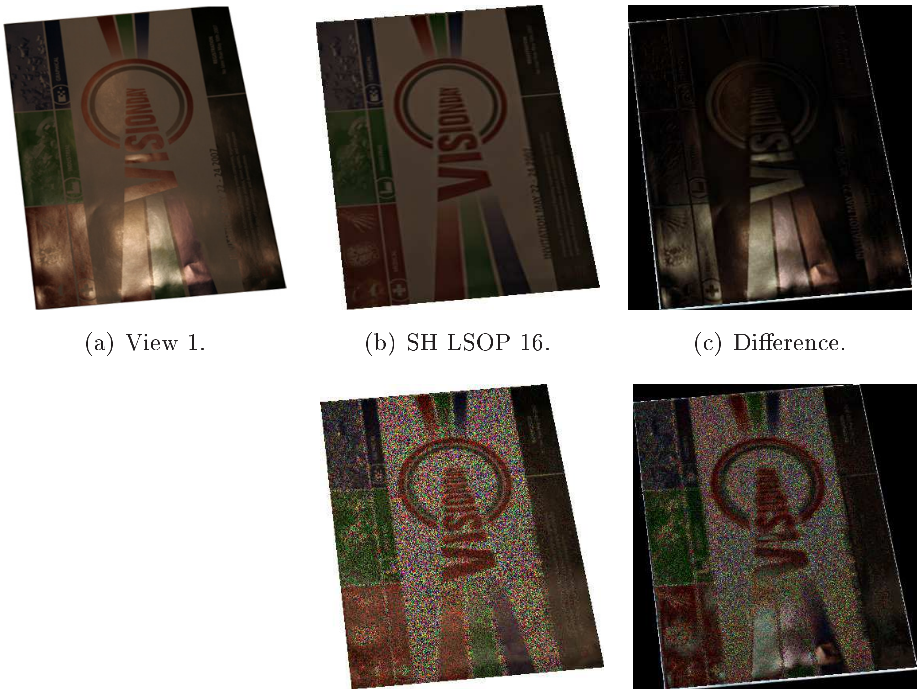
16.2.3 Comparison

To compare the results obtained using the three models, the mean and standard deviation of the difference images are shown Table 16.1. Due to the misalignment of the real and reconstructed image, the numbers are not completely accurate. However, they are still comparable, because they are made in a similar fashion.

It can be seen that the best results are achieved using the Phong model, where a mean value of 13 is measured. The SH ZH and LSOP with 16 parameters perform almost equally. However, when using 64 parameters the LSOP is very noisy and gives unusable results. For the second view where the surface is predominantly diffuse, all models are well-suited, with exception of the SH LSOP 64, where the noise is very clear.

16.2.4 Test Conclusion

The scene was well-suited for the Phong model, due to the appearance of the surface and amount and placement of light. That is the dominant reason why the Phong model proved



(a) View 1.

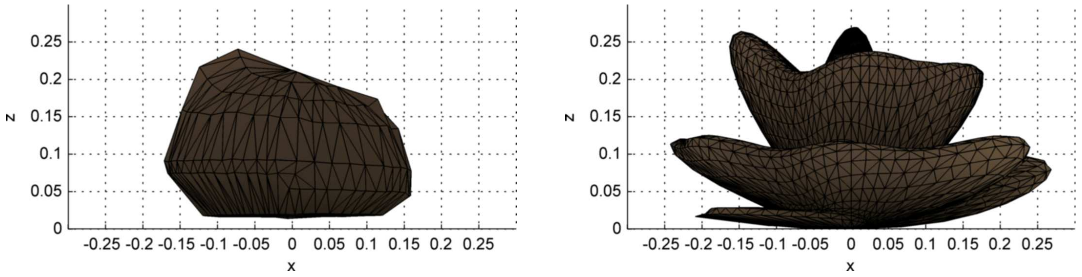
(b) SH LSOP 16.

(c) Difference.

(d) SH LSOP 64.

(e) Difference.

Figure 16.6: A comparison between the real and reconstructed flyer from the first view direction using 16 and 64 parameters in the SH LSOP model.



(a) Measured data from a sample point in the large white area.

(b) Reconstructed data from a sample point in the large white area.

Figure 16.7: Plots of measured and reconstructed data from a sample point in the large white area where noise is very distinct for the SH LSOP model with 64 parameters.

to be the best. However, it is not the only explanation. The SH ZH models were able to fit the diffuse parts of the surface acceptably with few parameters, but failed to model the highlights any good. If more parameters were used, the highlights were fitted better, but errors in the estimation were introduced. In general, the SH LSOP model performed quite bad. It seems that it required either more dense sampling of the view directions or some kind of pre- or postfiltering to counteract the very expressed ringing effects that caused the noise in the results.

Method:		**Phong**	**SHZH16**	**SHZH64**	**SHLSOP16**	**SHLSOP64**
View 1	Mean:	12.66	20.03	15.95	19.67	38.29
	Std.dev:	20.57	35.87	30.61	35.23	36.79
View 2	Mean:	3.66	4.17	6.52	4.05	22.12
	Std.dev:	12.71	12.97	13.50	12.74	31.50

Table 16.1: Overall mean values and standard deviations for the difference images obtained when comparing the real photographs with the reconstructed Visionday flyer using the different models.

16.3 Test: Patchwork

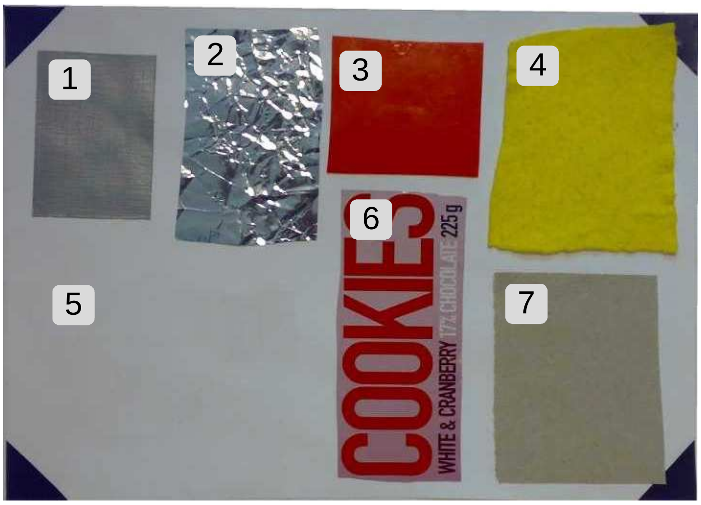
The patchwork surface seen in Figure 16.1(b) is comprised of numerous different materials. It is mounted on the same setup as in the previous test. The setup is shown in Figure 16.2 on page 102. Videos and images of this surface can be found on the DVD (⊚`/Video/TestVideos_Patchwork/`) and (⊚`/Test/figures/patchwork/`).

16.3.1 Test Introduction

Figure 16.8 shows a closeup of the surface with the different types of materials numbered and listed to the right. The different materials are chosen to cover a relatively wide range from diffuse to specular appearances. The kitchen cloth and paper towel patches are very diffuse, but the cloth have subtle details in its surface geometry and does not lay completely flat on the surface. The duct tape reflects light, but is not nearly as shiny as the plastic or the two types of foil. The two types of foil are furthermore very crumpled as they were creased before they were glued to the paper. The transparent tape is invisible from some angles, but from other angles it changes the perceived color of the paper and from other angles it reflects light.

To accommodate for the very shiny reflections from some of the types of material, the entire dynamic range of the surface must be captured. This is done by taking multiple photographs from each view direction with different exposure times and composing these images into one HDR image. Figure 16.9 sketches the concept of how HDR images are recorded in this test. The first image of the surface from a view direction is taken with a relatively high exposure time. If any of the pixels in the image are saturated, the exposure time is halved for the next image. This is repeated until none of the pixels in the image are saturated.

The ranges shown for the different pictures in the lower half of Figure 16.9 should be interpreted the following way. Saturated pixels in image 1 represents radiance values from one point and upwards on the radiance scale. Saturated pixels in image 2 represents radiances from a point higher on the scale and upwards. This way, less light is needed to saturate pixels in image 1 than in image 2 and so on. Thus, many images are needed to ensure no pixels are saturated for view directions close to the reflection direction and one



1:	Duct tape
2:	Aluminium foil
3:	Plastic
4:	Kitchen cloth
5:	Transparent tape
6:	Colored foil
7:	Paper towel

Figure 16.8: The patchwork surface comprised of several different materials.

image can be sufficient in the directions where no reflections are present. The maximum number of exposures needed during recording in this scene was eight, which was needed in two cases.

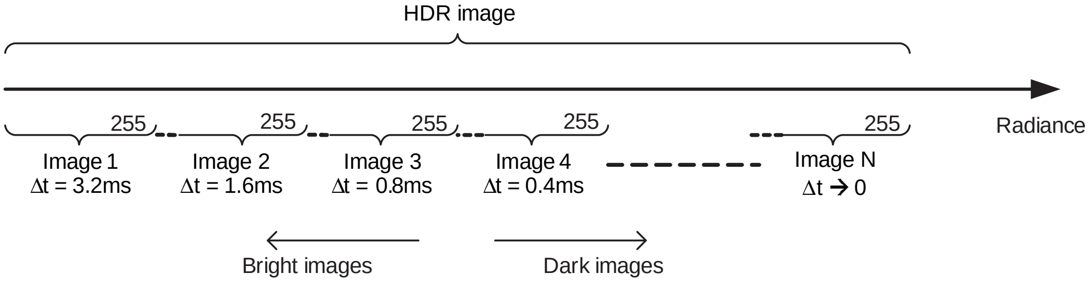


Figure 16.9: The concept of how HDR images are recorded in this test.

The goal is to compose an HDR image that holds the entire range of radiances in the scene from the current view direction. This involves converting the values of each of the images into radiance values. As accounted for in Chapter 7 on page 35, it is necessary to first find the response curve for the camera.

The two view directions that needed 8 images to cover their dynamic range were used to construct two response curves. As these curves were very similar, one of them were used to convert pixel values into radiances for all the recorded images. HDR Shop is used to recover the response curve, and compose the HDR images. It can make HDR images using a given response curve with different numbers of pictures. Two inputs are needed: exposure time steps and information regarding how each exposure relates to the darkest image used to construct the response curve.

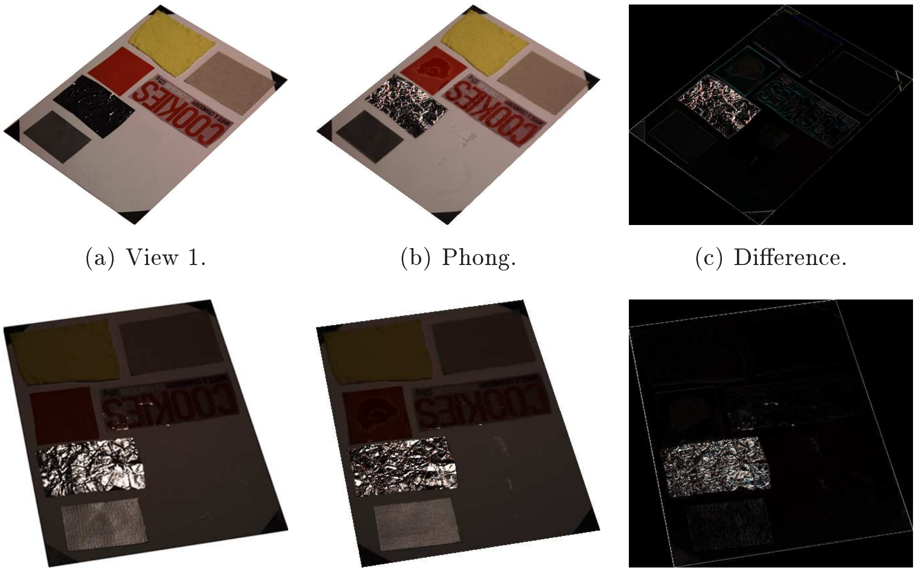
In total, the surface is sampled from 90 directions. First, 37 photographs were taken from the half of the hemisphere where no strong reflections are present. All these directions were sampled using only a single exposure. The other half of the hemisphere was sampled more densely, and multiple exposures were used where necessary.

16.3.2 Test Results

Both Phong, SH ZH and SH LSOP parameter maps are created for the patchwork surface. In the following, the results obtained for each of the models are presented. The results are shown for the same view directions for all models. Since the models are estimated using HDR images, the renderings and real images are tone mapped the same way for them to be comparable.

Phong

The results of the reconstruction of the patchwork using Phong are shown in Figure 16.10. The first column shows real images from two different directions. The second column shows the reconstructed surface. The third column depicts the difference images. The images for the two directions are shown with different exposures due to them covering different dynamic ranges.



(a) View 1. (b) Phong. (c) Difference.

(d) View 2. (e) Phong. (f) Difference.

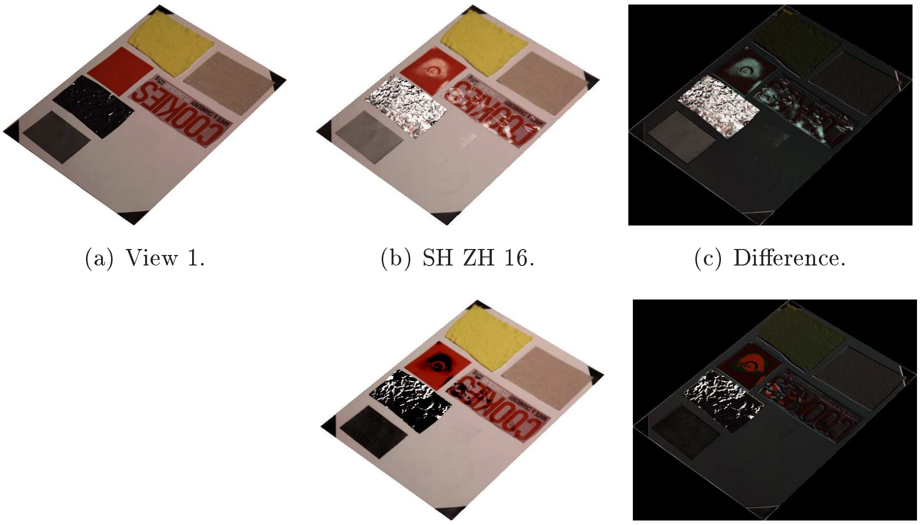
Figure 16.10: A comparison between the real and reconstructed patchwork using the Phong model for two view directions.

The diffuse radiance is modeled acceptably. This is noticeable at the kitchen cloth and paper towel. The same problems from the test of the Visionday flyer regarding very shiny materials are present. The highlight from the red plastic material is only sampled from a single direction where the highlight is only present on the middle of the patch. The result is a stain. The circle in the stain is due to the structure of the patch.

The errors in the foil are probably due to a slightly faulty estimation of the mirror direction. The highlight is only captured from few sample directions, which is why the fitted parameters might be slightly off.

SH ZH

The results of the reconstruction of the surface using SH ZH are shown in Figure 16.11. The view direction is the same as the first in the Phong test.



(a) View 1.

(b) SH ZH 16.

(c) Difference.

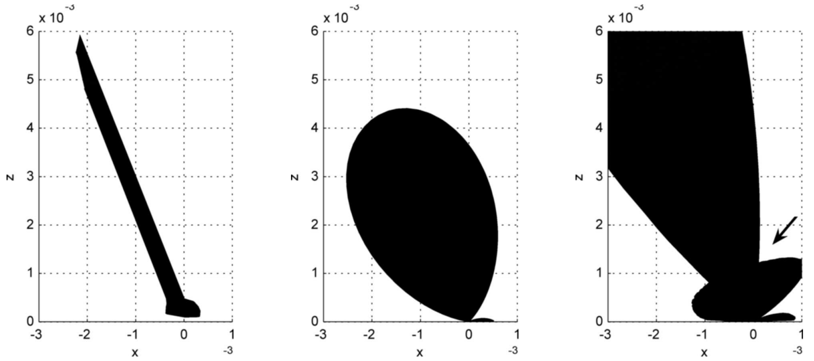
(d) SH ZH 64.

(e) Difference.

Figure 16.11: A comparison between the real and reconstructed patchwork from the first view direction using 16 and 64 parameters in the SH ZH model.

For 16 parameters, it can be seen that the strong highlights are modeled erroneously. They should not be visible from the reconstructed view direction. This is especially apparent on the plastic material. For 64 parameters the opposite is happening. Here, the points where a strong highlight has been sampled are estimated too low from this view direction. The highlight appears correct when seeing it from the reflection direction.

To investigate why the highlights appear wrongly from certain directions, the measured and reconstructed radiance distribution are plotted in Figure 16.12 for a point in the red patch using both 16 and 64 parameters. For 16 parameters, it is clear that the highlight is modeled to wide. Therefore the reconstructed surface appears white in Figure 16.11(b). From certain angle, the red patch will appear black as it does with more parameters. Using 64 parameters, the highlight is a bit more narrow. This causes the red plastic to appear black since it is reconstructed from a direction indicated by the arrow in Figure 16.12(c) around which the radiance values go towards zero.



(a) Measured data.

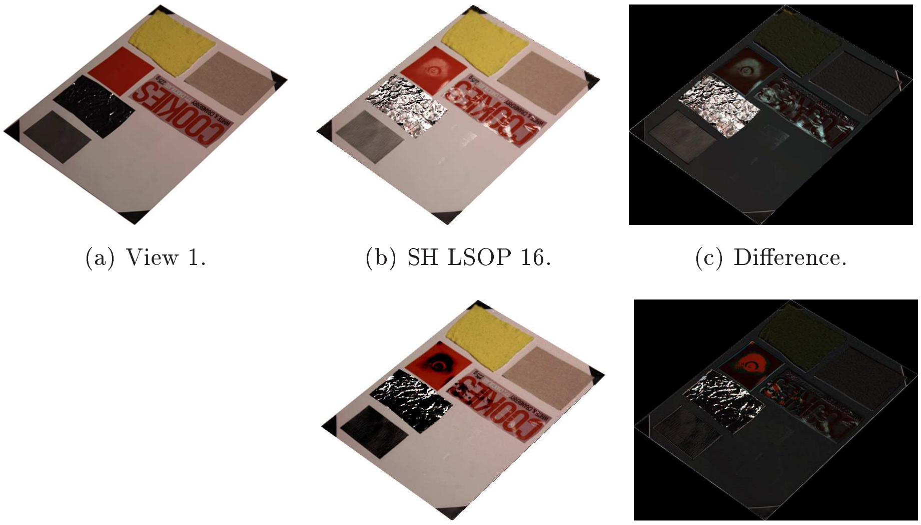
(b) Reconstructed data, 16 parameters.

(c) Reconstructed data, 64 parameters.

Figure 16.12: A sample point in the red plastic patch. The reconstructed highlight changes in a very unnatural manner. When the point is viewed from the direction indicated by the arrow, the point looks black, as the radiance goes towards zero.

SH LSOP

The results of the surface reconstructions using SH LSOP are illustrated in Figure 16.13. In the test, 16 and 36 parameters are used since reconstruction with 64 parameters proved to be very noisy in the Visionday test.



(a) View 1.

(b) SH LSOP 16.

(c) Difference.

(d) SH LSOP 64.

(e) Difference.

Figure 16.13: A comparison between the real and reconstructed patchwork from the first view direction using 16 and 36 parameters in the SH LSOP model.

The results using LSOP resembles the results achieved using ZH. Again, the shiny areas are reconstructed wrongly when seen from a direction that is not near the reflection direction.

16.3.3 Comparison

The results obtained using the three different models are summarized in Table 16.2. Again, the best results are achieved with the Phong model. It models both the highlights and diffuse parts with a small error compared to the SH models. In the test of the Visionday flyer, the SH models were able to model the diffuse reflection. However, this is not the case in this test. The mean values for the second and mostly diffuse view are very high compared to the ones obtained for the Visionday flyer. This is due to the very high range between the diffuse and specular part of the radiance distribution, as illustrated in Figure 16.12.

Another interesting feature of all the models is that they are able to model the phenomenon that the kitchen cloth does not lay completely flat on the surface as show in Figure 16.14. When viewing it from an oblique angle (16.14(a)), it can be seen that the lower, right edge of the cloth casts a shadow on the paper. When seeing it from behind, this shadow is not visible, and only the yellow appearance of the cloth is shown (16.14(b)). The fact that the surface in the application is indeed flat can be verified by looking at the cloth from a very oblique side view (16.14(c)).

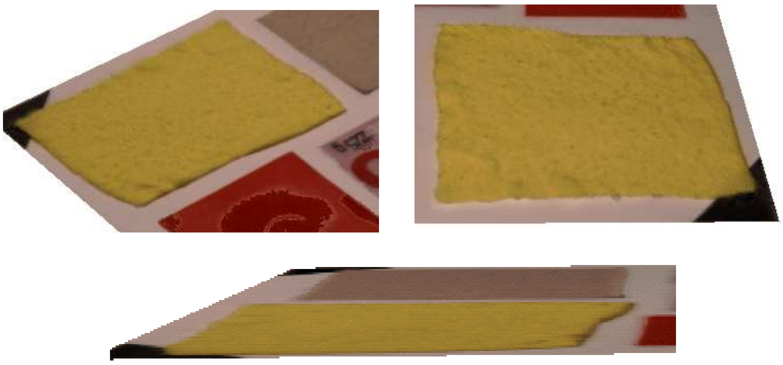


Figure 16.14: The model incorporates the phenomenon that the cloth does not lay completely flat on the paper and is able to recreate this by no use of extra geometry besides the flat quadrilateral.

Method:		**Phong**	**SHZH16**	**SHZH64**	**SHLSOP16**	**SHLSOP64**
View 1,	Mean:	5.03	20.22	15.33	21.61	12.54
$\gamma = 500$	Std.dev:	17.92	38.15	23.84	36.15	21.15
View 2,	Mean:	7.15	15.60	18.98	17.60	16.03
$\gamma = 100$	Std.dev:	24.31	32.16	31.73	32.87	32.90

Table 16.2: Overall mean values and standard deviations for the difference images obtained when comparing the real photographs with the reconstructed patchwork surface using the different models.

16.3.4 Test Conclusion

The conclusion on this test resembles the conclusions from the previous test to some extend. Generally the Phong model performed well, still due to the scene being quite well-suited for this model. The SH models performed worse by fitting the diffuse parts well with few parameters and the highlights better with more parameters, but not without introducing errors.

The surface was sampled using HDR images to capture its full dynamic range. The results were presented using a global tone mapping operator to allow for inspection of dark as well as bright areas. The use of HDR introduced some errors in the models. The highlights became very large and the SH models had trouble modeling that.

It can also be concluded that the highlights can easily become very large due to highly specular materials, as with the aluminium foil which is not modeled very precisely with any model. Another experience gained from this test is that highlights should be sampled quite densely in order to represented them sufficiently in the data that the models are fitted to. This is especially true for the large stain on the red plastic patch, which is a result of too sparse sampling of the highlight.

16.4 Summary

Different materials were tested to evaluate the feasibility of the methods and tools developed in this thesis on real-world data. The above tests show how well the models are able to reproduce the surface on the basis of the sampled data. The Phong model showed to be best, seen from an overall point of view. But it should be mentioned that the scene was quite well-suited for the employed Phong model. The more general SH model might be better suited for a scene with more complex lighting, which the Phong model is not able to fit.

It was also shown that the tools created and employed in this thesis are able to handle HDR sampling of a scene. However, it was a very cumbersome procedure and very error prone. Furthermore, SH had much trouble fitting the HDR data, because the highlights had values that were much larger than the diffuse values.

In general, the lesson learned is that much more dense sampling is needed. This is especially true if all highlight information is to be captured and incorporated into the models fitted to the data. However, the methods and tools in this thesis have proved their worth as good results were obtained from which much was learned.

17 Conclusion

After having evaluated the developed tools in the preceding chapter, this chapter concludes on the thesis as a whole. The results of the thesis are first summed up, before an answer to the initial problem is given.

This thesis set out to investigate the feasibility of capturing view-dependent radiance information of a surface. This was done with the aim of being able to compactly represent a precomputed global illumination solution of a surface in real time computer graphics applications. Instead of recreating a surface using a shading model and a number of light sources as normally done, the thesis seeks to directly represent the reflected radiances in all directions from the surface. This way, the surface can be reconstructed without explicitly modeling any light sources or material properties.

Initially, related work was analyzed in order find possible solutions to how the the initial problem can be solved. Two main approaches were found. The first approach was to represent surface radiances in their captured form, as images, and interpolate between these to create novel views of the surface. The second approach was to fit a parametric model to the measured radiances and evaluate the model when creating novel views. The second approach was chosen as it requires less storage when sampling a surface from a large number of directions. It was also chosen to store the parameters for the models in texture maps on a per point basis.

Based on the preliminary analysis, the main objective of this thesis was to build a framework to investigate the feasibility of sampling, modeling and recreating view-dependent radiance from a surface for use in a real time application. This problem statement consisted of three parts: sampling, modeling and recreation. Therefore, three goals were set for the thesis:

- Sample surfaces from different directions in a sufficient manner for later reconstruction.
- Capture the view-dependent characteristics of a surface and estimate parameters to models sufficient for describing them.
- Visualize the surface reconstructed at interactive frame rates in order to ensure that view-dependent properties of the surface have been captured and modeled correctly and sufficiently.

The sampling of the surface is the process of acquiring images of the surface from different view directions. To capture subtle details in the radiance emission from the surface, the surface must be sampled from a large number of directions. To determine the position of the camera in relation to the surface, camera calibration techniques are used. This is needed in order to find the outgoing direction of the measured radiances for each image. A surface can reflect light very differently in different directions. To handle this high range in the radiance emission, high dynamic range images of the surface are acquired by taking multiple images of the surface from the different sample directions. These images are assembled into one single image storing the captured radiances.

The acquired radiance emission for each point sampled on the surface requires a lot of storage. To reduce the needed storage, models are fitted to the measured data. Two different models were explored: One based on the Phong reflection model and another based on spherical harmonics. The Phong model requires very few parameters, but is only useful in a limited number of scenarios. It is able to model highlights on the surface very well under the given limitations. The spherical harmonics model required more parameters, but is a more general model which alleviates some of the limitations of the Phong model. The spherical harmonics model modeled the diffuse reflection of the surface well, but fails to represent strong highlights. The estimated parameters for the models are saved in texture maps.

By storing the estimated model parameters in textures, the power of modern graphics hardware is explored. Custom shader programs were written to reconstruct the appearance of the sampled surface using only the view direction as input. For both the applied models, it is possible to render the surface at interactive frame rates.

Based on tests on real data in the preceding chapter, it is concluded that the developed methods and tools are able to capture and approximate the surface radiance emission to some extend. Sampling is possible using both normal images and HDR images. However, the sampling process is very cumbersome since a large number of images is needed.

More work is required to model the measured radiances from a point on the surface accurately in all scenarios. The Phong model works well when assumptions about the lighting environment are true. However, this can not always be guaranteed. The more general representation with spherical harmonics is only able to model the diffuse radiance from the surface sufficiently. When using many parameters in the spherical harmonics model, the reconstructed surface appeared noisy. However, for cases where the acquired data is predominantly diffuse, the spherical harmonics are a good and compact representation.

It has been made possible to recreate the appearance of a real surface accurately, as shown in the tests, without explicitly modeling the light sources or the surface material. Only the emitted radiance from the surface is used. Because of this, the answer to the initiating problem is yes, it certainly seems feasible to capture the view-dependent radiance of a surface and reconstruct it in real time computer graphics applications.

18 Outlook

This chapter presents areas where further work are needed. It presents possible solutions to the issues encounter in the thesis which can be investigated to improve the results. Future work for sampling and modeling the radiance distribution from a surface are touched upon.

The first and most apparent area for further work is sampling. In the tests described in the first chapter of this part, it became evident that certain view directions need to be sampled more densely to capture the highlight better. It was learned that the sampling process is very cumbersome. It requires much time to ensure that the radiance distribution of all the points on the surface are sampled densely enough - and that their entire dynamic range is covered using photographs with different exposure times for the same view direction. It seems that it would prove worthwhile to invest some time in building an automated sampling rig, if more surfaces were to be sampled. The first step, though, would be to try and sample the surfaces mentioned in the last test more densely.

Another specific area that would draw benefit of more attention is the investigation of the least squares optimal projection of the spherical harmonics parameters. Lots of noise was introduced when 64 or more parameters were used, even though the highlights were more and more correctly approximated. The problem seems to be the introduction of ringing effects, a problem that might be counteracted by use of either pre- or post filtering of the acquired point radiance distributions. It could also be rewarding to try and fit the spherical harmonics model to data obtained in a more unconstrained environment, where not only one very expressed light source is present.

It could also prove advantageous to investigate other general models for fitting the acquired point radiance distributions. Comparative studies could reveal more details about the strengths and weaknesses of available methods. It could also be sought to implement algorithms that tried several models or variation of the same model on the same point data and selected the one that fitted best. This could e.g. be the employed Phong model with increasing numbers of light sources incorporated in the model until it fits the data well enough. More general shader programs that could take this into consideration would then need to be developed, but it does not seem impossible with the rapid development the field of custom shading development currently undergoes.

A set of tools for sampling, approximating and visualizing the view-dependent radiance of a surface have been developed in this thesis. These tools paves the way for much future research of the methods described in this thesis. There is even potential for becoming a standard way of representing realistic surfaces with subtle details in computer graphics applications where static scenes are acceptable. Though, this would require that robust methods are developed and automation of e.g. the sampling process is made possible.

Bibliography

[AB91] E. H. Adelson and J. R. Bergen. The Plenoptic Function and the Elements of Early Vision. *M. Landy and J. A. Movshon, (eds) Computational Models of Visual Processing*, 1991.

[Ang06] Edward Angel. *Interactive Computer Graphics.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 4nd edition, 2006.

[Arv95] James Richard Arvo. *Analytic Methods for Simulated Light Transport.* PhD thesis, Yale University, 1995.

[Bli78] James F. Blinn. Simulation of wrinkled surfaces. In *SIGGRAPH '78: Proceedings of the 5th annual conference on Computer graphics and interactive techniques*, pages 286–292, New York, NY, USA, 1978. ACM Press.

[BN76] James F. Blinn and Martin E. Newell. Texture and reflection in computer generated images. *Commun. ACM*, 19(10):542–547, 1976.

[Bou07] Jean-Yves Bouguet. Camera calibration toolbox for matlab. `http://www.vision.caltech.edu/bouguetj/calib_doc/`, 2007.

[Cat75] Edwin E. Catmull. Computer display of curved surfaces. In *Proceedings of the IEEE Conference on Computer Graphics, Pattern Recognition, and Data Structure*, pages 11–17, 1975.

[CL96] Thomas F. Coleman and Yuying Li. An interior trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on Optimization*, 6(2):418–445, 1996.

[CMSR98] P. Cignoni, C. Montani, R. Scopigno, and C. Rocchini. A general method for preserving attribute values on simplified meshes. In *VIS '98: Proceedings of the conference on Visualization '98*, pages 59–66, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press.

[COM98] Jonathan Cohen, Marc Olano, and Dinesh Manocha. Appearance-preserving simplification. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 115–122, New York, NY, USA, 1998. ACM Press.

[DBY98] Paul E. Debevec, George Borshukov, and Yizhou Yu. Efficient view-dependent image-based rendering with projective texture-mapping. In *9th Eurographics Rendering Workshop,*, 1998.

[DM97] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. *Computer Graphics*, 31(Annual Conference Series):369–378, 1997.

[DvGNK97] Kristin J. Dana, Bram van Ginneken, Shree K. Nayar, and Jan J. Koenderink. Reflectance and texture of real-world surfaces. *IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[Eli07] Hugo Elias. Exposure. `http://freespace.virgin.net/hugo.elias/graphics/x_posure.htm`, 2007.

[GGSC96] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 43–54, New York, NY, USA, 1996. ACM Press.

[Gre03] Robin Green. Spherical harmonic lighting: The gritty details. Technical report, Sony Computer Entertainment America, 2003.

[GW02] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing.* Prentice Hall, 2002.

[Han04] Pat Hanrahan. Computer graphics: Image synthesis techniques. `http://graphics.stanford.edu/courses/cs348b-04/lectures/lecture15/`, 2004. Course CS 348B.

[JMLH01] Henrik Wann Jensen, Stephen R. Marschner, Marc Levoy, and Pat Hanrahan. A practical model for subsurface light transport. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 511–518, New York, NY, USA, 2001. ACM Press.

[Kaj86] James T. Kajiya. The rendering equation. In *SIGGRAPH '86: Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150, New York, NY, USA, 1986. ACM Press.

[KTI$^+$01] Tomomichi Kaneko, Toshiyuki Takahei, Masahiko Inami, Naoki Kawakami, Yasuyuki Yanagida, Taro Maeda, and Susumu Tachi. Detailed shape representation with parallax mapping. In *The 11th International Conference on Artificial Reality and Telexistence*, Tokyo, 2001.

[KvDS96] Jan J. Koenderink, Andrea J. van Doorn, and Marigo Stavridi. Bidirectional reflection distribution function expressed in terms of surface scattering modes. In *ECCV '96: Proceedings of the 4th European Conference on Computer Vision-Volume II*, pages 28–39, London, UK, 1996. Springer-Verlag.

[LFTG97] Eric P. F. Lafortune, Sing-Choong Foo, Kenneth E. Torrance, and Donald P. Greenberg. Non-linear approximation of reflectance functions. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 117–126, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[LH96] Marc Levoy and Pat Hanrahan. Light field rendering. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31–42, New York, NY, USA, 1996. ACM Press.

[LKG+01] Hendrik P. A. Lensch, Jan Kautz, Michael Goesele, Wolfgang Heidrich, and Hans-Peter Seidel. Image-based reconstruction of spatially varying materials. In Steven Gortler and Karol Myszkowski, editors, *Rendering Techniques 2001: Proceedings of the 12th Eurographics Workshop on Rendering*, pages 104–115, London, Great Britain, 2001. Springer.

[MRP98] Gavin S. P. Miller, Steven M. Rubin, and Dulce B. Ponceleon. Lazy decompression of surface light fields for precomputed global illumination. In *Rendering Techniques*, pages 281–292, 1998.

[Nic76] F. E. Nicodemus. Self-study manual on optical radiation measurements: Part i–concepts, chapters 1 to 3. Technical report, National Bureau of Standards, 1976. Technical Note 910–1.

[NLC02] Hyeon-Suk Na, Chung-Nim Lee, and Otfried Cheong. Voronoi diagrams on the sphere. *Comput. Geom. Theory Appl.*, 23(2):183–194, 2002.

[NRH+77] F. E. Nicodemus, J. C. Richmond, J. J. Hsia, I. W. Ginsberg, and T. Limperis. Geometrical considerations and nomenclature for reflectance. Technical report, National Bureau of Standards, 1977. Technical Note 910–1.

[NRH03] Ren Ng, Ravi Ramamoorthi, and Pat Hanrahan. All-frequency shadows using non-linear wavelet lighting approximation. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 376–381, New York, NY, USA, 2003. ACM Press.

[nVI04] nVIDIA. Improve batching using texture atlases. 2004.

[ON95] M. Oren and S.K. Nayar. Generalization of the lambertian model and implications for machine vision. *IJCV*, 14(3):227–251, April 1995.

[OS89] Alan V. Oppenheim and Ronald W. Schafer. *Distrete-Time Signal Processing.* Prentice Hall, 1989.

[Pho75] Bui Tuong Phong. Illumination for computer generated pictures. *Commun. ACM*, 18(6):311–317, 1975.

[RH01] Ravi Ramamoorthi and Pat Hanrahan. An efficient representation for irradiance environment maps. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 497–500, New York, NY, USA, 2001. ACM Press.

[Ros06] Randi J. Rost. *OpenGL Shading Language*. Addison-Wesley Longman Publishing Co., Inc., 2nd edition, 2006.

[RWPD06] Erik Reinhard, Greg Ward, Sumanta Pattanaik, and Paul Debevec. *High Dynamic Range Imaging*. Morgan Kaufmann, 2006.

[SAG+05] Peter Shirley, Michael Ashikmin, Michael Gleichner, Stephen R. Marschner, Erik Reinhard, Kenvin Sung, William B. Thompson, and Peter Willemsen. *Fundamentals of Computer Graphics*. A K Peters Ltd., 2nd edition, 2005.

[SHHS03] Peter-Pike Sloan, Jesse Hall, John Hart, and John Snyder. Clustered principal components for precomputed radiance transfer. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 382–391, New York, NY, USA, 2003. ACM Press.

[SHR+99] H. Schirmacher, W. Heidrich, M. Rubick, D. Schiron, and H. Seidel. Image-based brdf reconstruction, 1999.

[SKS02] Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *Proceedings of ACM SIGGRAPH 2002 (SIGGRAPH-02)*, volume 21 of *acm Transactions on Graphics*, pages 527–536, San Antonio, USA, July 2002. Association of Computing Machinery (ACM), ACM.

[Sto] Duane Storey.
hdr tutorial. ://www.duanestorey.com/hdr-tutorial/.

[SvBLD03] Frank Suykens, Karl vom Berge, Ares Lagae, and Philip Dutré. Interactive rendering with bidirectional texture functions. *Computer Graphics Forum*, 22(3), 2003.

[SWND06] Dave Shreiner, Mason Woo, Jackie Neider, and Tom Davis. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 2*. Addison-Wesley Longman Publishing Co., Inc., 5th edition, 2006.

[TV98] Emanuele Trucco and Alessandro Verri. *Introdutory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.

[WAA+00] Daniel N. Wood, Daniel I. Azuma, Ken Aldinger, Brian Curless, Tom Duchamp, David H. Salesin, and Werner Stuetzle. Surface light fields for 3D photography. In Kurt Akeley, editor, *Siggraph 2000, Computer Graphics Proceedings*, pages 287–296. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.

[War92] Gregory J. Ward. Measuring and modeling anisotropic reflection. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 265–272, New York, NY, USA, 1992. ACM Press.

[WAT92] Stephen H. Westin, James R. Arvo, and Kenneth E. Torrance. Predicting reflectance functions from complex surfaces. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 255–264, New York, NY, USA, 1992. ACM Press.

[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 *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 311–320, New York, NY, USA, 1987. ACM Press.

[ZC04] Cha Zhang and Tsuhan Chen. A survey on image-based rendering – representation, sampling and compression. In *Signal Processing: Image Communication*, volume 19, pages 1–28. Elsevier, 2004.

Part VI
Appendices

This part contains a number of appendixes that support the thesis with more details on certain subjects.

The first appendix explains and details two relevant geometric topics, "spherical coordinates" and "solid angles".

The second appendix details on the subject of colorimetry, how colors are represented and recorded.

The third appendix presents basic radiometry, which is the science of propagation of light and measurement thereof. The concept of light, how it behaves and how it can be recorded and reproduced is described.

The next appendix details upon the Bidirectional Reflectance Distribution Function, a ubiquitous computer graphics term which is a 4-dimensional function describing the ratio of reflected to incident radiance for a surface.

An appendix is also given that presents the "rendering equation", an equation that expresses all interaction between light sources and surfaces as well as reflection surfaces in between.

The last technical appendix presents the technique of camera calibration that is employed in this thesis in a general fashion.

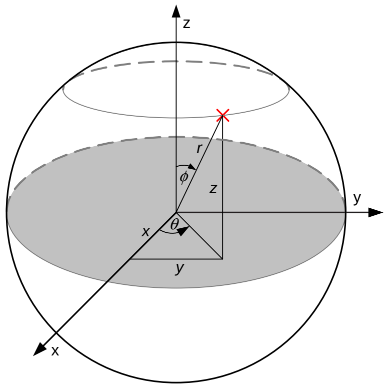
The last appendix describes the content of the enclosed DVD and gives instructions on how to use the supplied application.

A Mathematical Preliminaries

This appendix defines two relevant geometric topics: spherical coordinates and solid angles. The following sections provide a review of these topics, to act as a basis for using them in the thesis.

A.1 Spherical Coordinates

Usually, the position of a point is given in cartesian coordinates, be it $[x, y]$ in $\mathbb{R}^2$, $[x, y, z]$ in $\mathbb{R}^3$ etc. However, when dealing with circles or spheres, it can be convenient to express the position of the point in polar coordinates. In $\mathbb{R}^3$, when dealing with a sphere, these coordinates are called “spherical coordinates”.



$$
\begin{aligned}
r &= \sqrt{x^2 + y^2 + z^2} \\
\theta &= tan^{-1}\left(\tfrac{y}{x}\right) \\
\phi &= cos^{-1}\left(\tfrac{z}{r}\right)
\end{aligned}
$$

$$
\begin{aligned}
x &= r \cdot cos(\theta) \cdot sin(\phi) \\
y &= r \cdot sin(\theta) \cdot sin(\phi) \\
z &= r \cdot cos(\phi)
\end{aligned}
$$

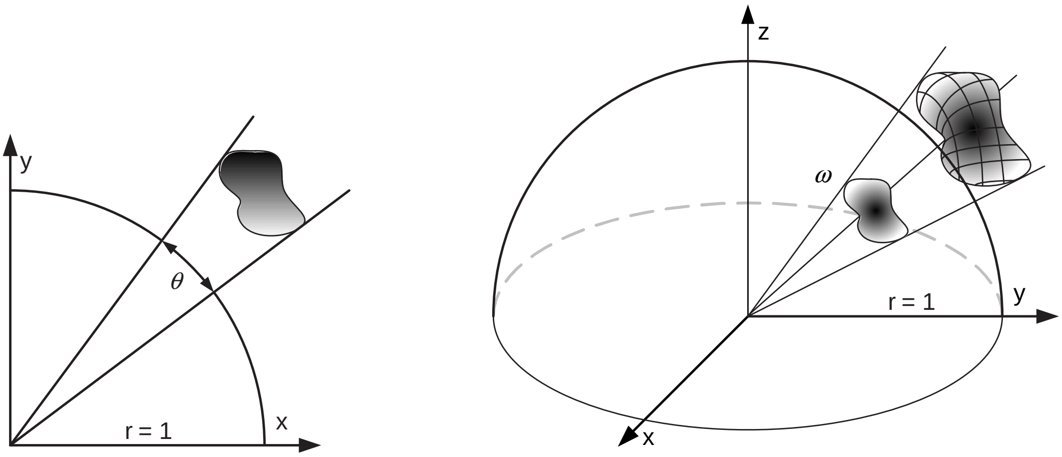
Figure A.1: A visualization of coordinates on a sphere. A point on the sphere, marked with a red cross, is shown in both cartesian coordinates, $[x, y, z]$, and spherical coordinates, $[r, \theta, \phi]$.

The left part of Figure A.1 shows a sphere with a point on its surface. Its x-, y- and z-coordinates can be converted to spherical coordinates, so the point is expressed using the radius of the sphere, $r \in [0; \infty[$, the azimuthal angle, $\theta \in [0; 2\pi[$, measured in the xy-plane from the x-axis and the zenithal angle, $\phi \in [0; \pi]$, measured from the positive z-axis. The relationship between cartesian and spherical coordinates in $\mathbb{R}^3$ is given in the right part of the figure.

$[x, y, z]$ and $[r, \theta, \phi]$ both represent a point on an arbitrary large sphere in $\mathbb{R}^3$. If a normalized direction from the origin to the point is needed, one can either normalize the $[x, y, z]$ vector or simply omit the r coordinate to obtain a more compact representation, $[\theta, \phi]$.

A.2 Solid Angles

This section on solid angles is based on [Nic76]. Solid angles can be related to angles as known from measuring on a 2D surface. Figure A.2(a) shows the angle θ subtended on a unit circle by an object. The 3D counterpart is the solid angle ω subtended by the object onto the unit sphere, as shown in Figure A.2(b).



(a) Angle measured on a circle.

(b) Solid angle measure on a sphere.

Figure A.2: Figures showing an object subtending an angle on a circle in 2D and a solid angle on a sphere in 3D.

An angle in 2D is normally measured in degrees, or in radians. Radians $[rad]$ express the length of arc that the angle spans on a unit circle. Solid angles are measured in steradians, which analogously to a length of an arc, express an area on the surface of a unit sphere. Thus steradians $[sr]$ equal radians squared, $[rad^2] = [sr]$.

To define this in a more mathematical manner, solid angles can be derived from differential solid angles. Figure A.3 shows a differential solid angle as the red patch on the surface on a sphere. For angles of θ and ϕ with small increments $\Delta\theta$ and $\Delta\phi$ going towards zero, the area goes towards the differential solid angle:

$$\lim_{\Delta\phi,\Delta\theta\to 0} sin(\phi)\cdot\Delta\theta\cdot\Delta\phi = d\omega = sin(\phi)\cdot d\theta\cdot d\phi$$

This is the area of the patch in Figure A.3, where $d\phi$ represents the length of the vertical sides, while $sin(\phi)\cdot\theta$ represents the length of horizontal sides, with compensation for the curvature of the sphere surface. Thus $d\omega$ defines a differential solid angle in units of steradians, $[sr]$

To extend this concept to solid angles in general, an integration over all directions $(\theta,\phi)\in\mathcal{A}$ pointing towards the area of ω is performed to obtain a value for ω:

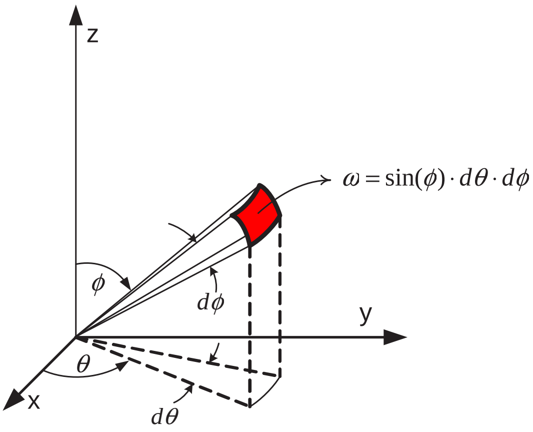


Figure A.3: A differential solid angle shown as a red patch on a unit sphere.

$$\omega = \int_{\mathcal{A}} d\omega = \int_{\mathcal{A}} sin(\phi) d\theta d\phi$$

To exemplify, the surface area of a unit sphere is known to be 4π. This can be verified using solid angles by integrating over the entire sphere of directions, $\mathcal{A} = \mathcal{H}^4$, where $0 \leq \phi \leq \pi$ and $0 \leq \theta < 2\pi$:

$$\omega_{\mathcal{H}^4} = \int_{\phi=0}^{\pi} \int_{\theta=0}^{2\pi} sin(\phi) d\theta d\phi = \int_{\phi=0}^{\pi} 2\pi \cdot sin(\phi) d\theta = [-2\pi \cdot cos(\phi)]_0^{\pi} = 2\pi - (-2\pi) = 4\pi$$

Thus, there are 4π steradians in a sphere, and likewise 2π steradians in a hemisphere ($\mathcal{H}^2$) and so on.

B Colorimetry

As the real world presents itself using a variety of colors, some basic knowledge about colors is presented in this appendix. It is furthermore exemplified how a camera can capture colors for later reproduction, e.g. as an image on a computer screen.

The field of colorimetry is concerned with assigning numbers to physical stimuli, such that other stimuli with equal specifications look alike [RWPD06]. One of the most prevalent findings from colorimetry experiments are the three primary stimuli, which can be combined to match almost all colors. For each color the three primaries may be adjusted and combined to create a match.

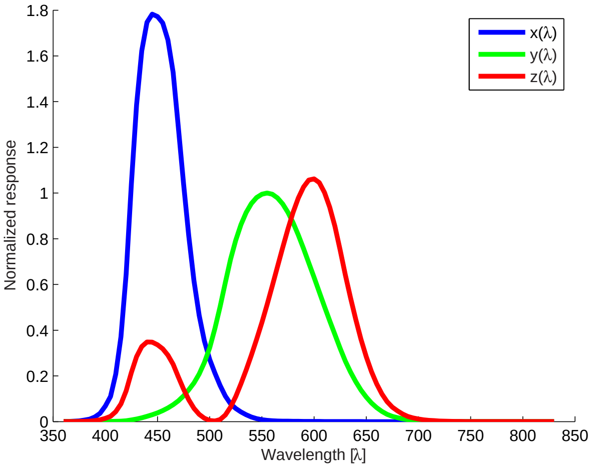


Figure B.1: CIE 1931 2-degree XYZ color matching functions. They represent the "CIE 1931 standard observer". (Data downloaded from `http://www.cvrl.org`).

Figure B.1 shows what is called the color matching functions for the "CIE 1931 standard observer". They are obtained by measuring the intensities of three primaries at different wavelengths, [λ]. A stimulus color Q_λ can then be represented as a linear combination of these three functions:

$$Q_\lambda = x(\lambda) \cdot X + y(\lambda) \cdot Y + z(\lambda) \cdot Z$$

Thus, the (X, Y, Z) scalar multipliers are the tristimulus values of a color Q_λ. They are analogous to an (R, G, B)-triplet known from imaging, but denoted (X, Y, Z) because the color matching functions are fixed to always take positive tristimulus values for visible colors [RWPD06].

The observed color of an object depends roughly on three things, if the media through which it propagates is disregarded; the light source, the surface and the observer. Figure B.2 shows a visualization of this concept. The observer has a response curve as in Figure B.1. The light source has an equivalent curve, representing the light it emits, and the surface has a curve for the light it reflects (absorbed light is disregarded). In this example, the light is primarily red, the surface reflects blue light the most and green and red light slightly less. The camera is a standard observer, capable of recording visible colors.

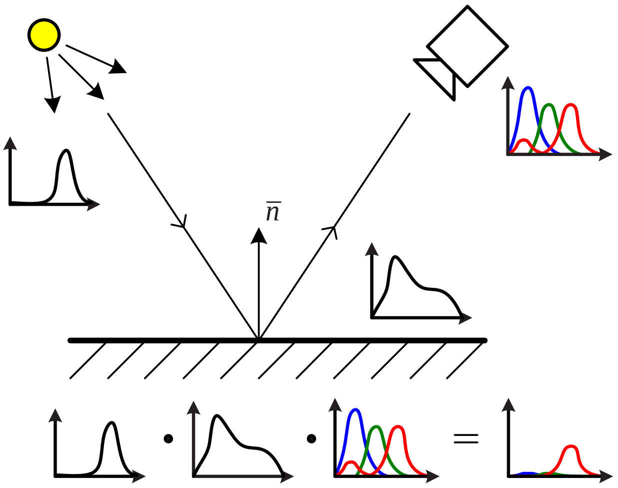


Figure B.2: A model of how light is transported from a light source, reflected on a surface and observed by a camera. The recorded color is the product of all three curves, a red depiction of the surface.

The final color in the example in Figure B.2 is the product of the three curves. The end result, as indicated in the bottom of the figure, is a red depiction of the surface.

The way that the light is reflected from the surface, depends on a number of properties for that particular surface. Color is one of them, along with information about how the surface reflects and absorbs light in general. These two subjects are the topics of the following two appendices.

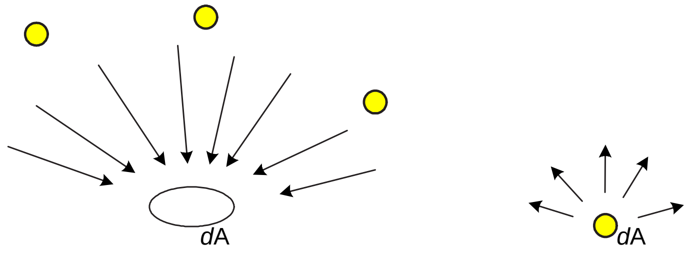
C

Radiometry and Light

Capturing an image, be it in digital or analogue form, is basically the process of exposing a light sensitive chip or film to light in a short amount of time. The light that hits the chip or film during that time frame, constitutes an image. A number of measurable quantities for light are presented and clarified in this appendix, to gain an understanding of how light behaves and can be recorded and reproduced.

Radiometry is the science of propagation of light and measurement thereof. The primary source for this appendix is [RWPD06], from which the previous definition is taken. Light is radiant energy, measured in Joules, propagating through different media such as air, water etc. As light travels through space, a measurable flow of radiant energy is present. As this is measured in Joules per second, it is a measure of energy per unit of time [W].

The radiant flux hitting a point is known as irradiance, which is exemplified in Figure C.1(a). The irradiance is radiant flux from all directions arriving at an area on a surface. Reversely, radiant exitance is the radiant flux leaving a point in all directions, exemplified in Figure C.1(b). Both these quantities are measured in watts per area [Wm^{-2}] and the difference lies in the domain of integration, whether it is the incident hemisphere (irradiance) or the outgoing hemisphere (radiant exitance).



(a) Irradiance: power incident upon a unit area dA.

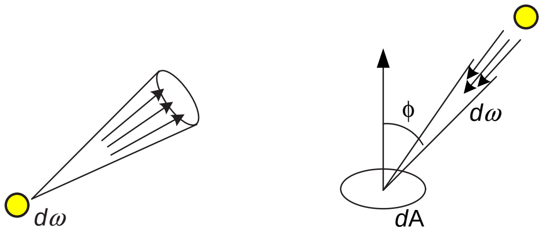
(b) Radiant exitance: power emitted per unit area.

Figure C.1: Figures showing the concepts of irradiance and radiant exitance. Figure after [RWPD06].

If an infinitesimally small point light source is considered, the amount of light emitted in a particular direction is called radiant intensity. It is measured in watts per steradian, [Wsr^{-1}]. Figure C.2(a) shows this concept.

The radiant flux incident at a point on a surface from a particular direction is known as radiance. Radiance is the physical counterpart to the physiological concept of brightness or intensity [Arv95]. It is a measure of energy per unit of time, area and direction. Thus the unit is watts per area per steradian, [$Wm^{-2}sr^{-1}$]. The concept of radiance is shown in Figure C.2(b). However, it should be noted that radiance is not limited to light entering an area as shown in the figure. According to the definition in [RWPD06], radiance is flux passing through, leaving or arriving at a point in a particular direction.

Light hitting a point on a surface from a particular direction is the heart of image formation. The combination of shutter, lens and sensor in a camera, restricts light in such a fashion, that when taking an image, each pixel is essentially measuring the incoming radiance.



(a) Radiant intensity: power emitted per solid angle $d\omega$.

(b) Radiance: power incident on a unit area dA from a unit set of directions $d\omega$.

Figure C.2: Figures showing the concepts of radiant intensity and radiance. Figure after [RWPD06].

A summary of all radiometric quanities is given in Table C.1.

Radiometric quantity	Unit
Radiant Energy	J
Radiant Power	W
Irradiance	Wm^{-2}
Radiant Exitance	Wm^{-2}
Radiant Intensity	Wsr^{-1}
Radiance	$Wm^{-2}sr^{-1}$

Table C.1: Radiometric quantities

Light hitting a surface is either absorbed and converted to heat or reflected in some directions. The appearance of the surface is dependent on the how the light is reflected. A matte surface reflects light almost evenly in all directions whereas a glossy surface will reflect light strongly in certain directions. The interaction between light and a material can be modeled using the BRDF, which is the topic of the following appendix.

D The Bidirectional Reflectance Distribution Function

This appendix describes the BRDF, a ubiquitous term within the field of computer graphics. It is derived and presented as a ratio of reflected to incident radiance, and a way of functionally describing how light reflects off of a surface.

The BRDF is a function that described the distribution of reflectance based on two directions. *Reflection* is the process of flux hitting a surface and leaving it again on the same side, while *reflectance* is the fraction of incident flux that is reflected. First, the definition of a BRDF is presented based on [NRH$^+$77]. Then some important properties are listed and described.

D.1 Definition

A BRDF shows a relation of irradiance from one given direction to its contribution to the reflected radiance in another direction. Purely diffuse and purely specular reflection can be quite closely approximated, but neither appears completely in practice. The usual practice is therefore to present the reflectance of a surface by a number of specular highlights and a diffuse component, because of their ability to be linearly combined, as exemplified at a later point in this appendix.

Only parametric BRDFs are considered in this appendix with the assumption that light enters and leaves the surface at the same point as stated in [JMLH01] for valid BRDFs. This is the outcome of the BRDF being a simplification of the more general Bidirectional Surface Scattering Reflection Distribution Function (BSSRDF), which describes the relationship between any two rays hitting a surface. Almost all BRDFs are derived from surface scattering, with any sub-surface scattering being approximated by a Lambertian (diffuse) component.

Let the portion of incident flux from a direction (θ_i, ϕ_i) striking an area, dA_i centered at the point (x_i, y_i) be denoted $d\Phi_i$. The reflected radiance in the direction (θ_r, ϕ_r) at the point (x_r, y_r) coming from $d\Phi_i$ is denoted dL_r. Generally, dL_r is proportional to $d\Phi_i$, so

$$dL_r = S \cdot d\Phi_i, \tag{D.1}$$

with subscript i for incident light and subscript r for reflected light. In this case, S is an

8-dimensional function, $S = S(\theta_i, \phi_i, x_i, y_i, \theta_r, \phi_r, x_r, y_r)$.

If it is assumed that the area dA_i is uniformly irradiated, the incident radiance depends only on the direction, (θ_i, ϕ_i). The incident flux then becomes:

$$d\Phi_i = L_i(\theta_i, \phi_i) \cdot cos(\phi_i) \cdot d\omega_i \cdot dA_i = dE_i \cdot dA_i \tag{D.2}$$

where dE_i is the incident irradiance and $d\omega_i$ is the solid angle in which the irradiance is confined. The contributions to reflected radiance from all the incident flux in the direction (θ_i, ϕ_i) and within the solid angle $d\omega_i$ can be found by integrating over the area A_i (by combination of Equation D.1 and D.2):

$$dL_r = \int_{A_i} S \cdot d\Phi_i = dE_i \cdot \int_{A_i} S \cdot dA_i \tag{D.3}$$

If it is further assumed that the scattering properties of the surfaces are uniform and isotropic, Equation D.3 can be written as

$$dL_r = dE_i \cdot f_r(\theta_i, \phi_i, \theta_r, \phi_r) \quad \text{,where} \quad f_r = \int_{A_i} S(\theta_i, \phi_i, \theta_r, \phi_r) \cdot dA_i$$

Thus, the characterization properties of radiance over a uniform and isotropic surface, f_r, is:

$$f_r(\theta_i, \phi_i, \theta_r, \phi_r) = \frac{dL_r}{dE_i} = \frac{dL_r(\theta_i, \phi_i, \theta_r, \phi_r)}{L_i(\theta_i, \phi_i) \cdot cos(\phi_i) \cdot d\omega_i}$$

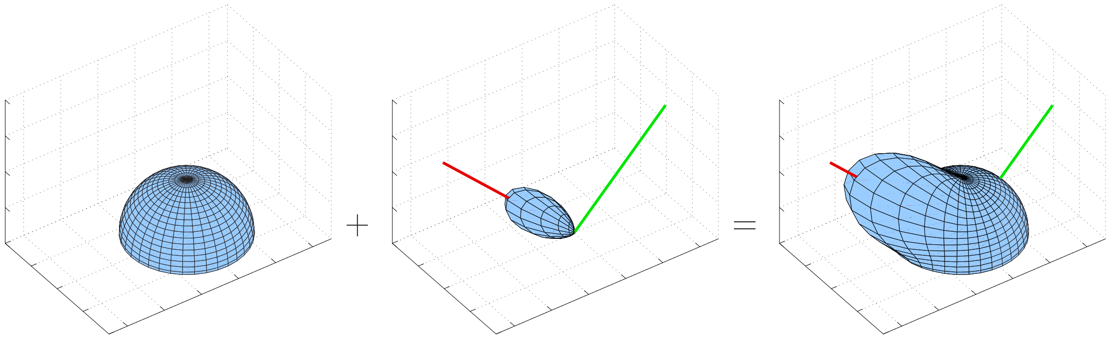
the four-dimensional BRDF. It describes the ratio of reflected light over incoming light. It is a ratio of infinitesimals and a derivative with “instantaneous” values that can never be measured directly.

D.2 Properties

A number of properties exist for the BRDF. Four of these are exemplified in the following paragraphs.

The BRDF must adhere to the conservation of energy laws. As reflectance is a ratio of reflected to incident flux it can only be within the interval $[0; 1]$. Accordingly, the BRDF, being a concentration of reflectance, can therefore theoretically attain values from $[0; \infty[$.

The second property is linearity, meaning that several BRDFs can be linearly combined to one single BRDF. This principle is often used in rendering to combine specular and diffuse BRDFs into one. It is important that the sum of the reflectances of the BRDFs must be less than or equal to 1, to adhere to the first property. Figure D.1 shows a pictorial example of linear combination of BRDFs.



(a) Diffuse component. (b) Specular component. (c) Resulting BRDF.

Figure D.1: The resulting Phong-like BRDF is a linear combination of the diffuse and specular BRDFs. The green line indicates direction from point to light, and the red line indicates reflection direction.

Reciprocity is another important property of the BRDF, meaning that it does not matter if the incident and radiant directions are switched, the end result is the same:

$$f_r(\theta_1, \phi_1, \theta_2, \phi 2) = f_r(\theta_2, \phi_2, \theta_1, \phi 1)$$

When discussing reflectivity of surfaces, isotrophy and anisotrophy are two apparent concepts to come up. A BRDF is denoted isotropic if its value remains the same when the surface is rotated around its normal at the point of current interest. It is thus possible to reduce the dimensionality of the BRDF with one:

$$f_r(\theta_i, \phi_i, \theta_r, \phi_r) \rightarrow f_r(\theta_r - \theta_i, \phi_i, \phi_r)$$

An example of an isotropic BRDF is the well-known Phong specular reflection model [Pho75]. Reversely, anisotropic BRDFs change the appearance of the surface when it is rotated around its normal. These are normally used for modeling e.g. hair and brushed metal. An example of an anisotropic BRDF is the Ward BRDF [War92].

E The Rendering Equation

For completeness when lighting surfaces of a scene, all interaction between light sources and surfaces as well as reflection surfaces in between needs to be taken into consideration. This is why the topic of this appendix is the rendering equation, and an example showing the difference between the terms "global" and "local' illumination is given.

The rendering equation is a general way of expressing balance in energy flow between points on surfaces. It has its foundation in energy balance, and seeks to abide to the following equilibrium of light [Han04]:

$$[\text{outgoing}] - [\text{incoming}] = [\text{emitted}] - [\text{absorbed}]$$

Rephrasing this means that all outgoing light from a point equals the sum of all incoming light and the light it emits, minus the light it absorbs.

The rendering equation shown below was presented in [Kaj86], and is stated without any assumptions about reflectance properties of the surfaces involved:

$$I(x, x') = g(x, x') \cdot \left[\epsilon(x, x') + \int_S \rho(x, x', x'') \cdot I(x', x'') dx'' \right], \text{ where:}$$

- $I(x, x')$ is the intensity of the light transported from point x' to x.
- $g(x, x')$ is a geometry term expressing visibility and distance between the points in question.
- $\epsilon(x, x')$ is an expression of emitted light from point x' to x.
- $\rho(x, x', x'')$ expresses the intensity of scattered light from x'' to x by a surface at x'.
- $\int_S$ is an integral over the union of all surfaces, thus the points x, x' and x'' range over all surfaces on all objects in the scene.

Rewriting this using equations derived in Appendix D and including the position on the surface as $\overline{x}$, the rendering equation can be expressed as:

$$L_r(\overline{x}, \theta_o, \phi_o) = L_e(\overline{x}, \theta_o, \phi_o) + \int_{\mathcal{H}} f_r(\overline{x}, \theta_i, \phi_i, \theta_o, \phi_o) \cdot L_i(\overline{x}, \theta_i, \phi i) \cdot cos(\phi_i) \cdot d\omega_i$$

where $L_e(\overline{x}, \theta_i, \phi_i)$ expresses the amount of light that the surface emits in the outgoing direction at the respective point.

E.1 Approximate Solutions

Solving the rendering equation would result in a correctly lit scene, taking all possible ways of lighting surfaces into account. A complete solution requires an infeasible amount of processing, and therefore a number of approximate solutions exist. Two of the most prevalent are radiosity and ray tracing. These are shortly introduced as a conclusion to this appendix.

Radiosity Radiosity originates from analysis of heat transfer between surfaces in enclosed environments. It allows the radiant energy intensity arriving at a surface to be computed. These intensities can then be used for shading the surface. Radiosity operates with the assumption that all surfaces emit and reflect energy diffusely and uniformly over the entire surface. It also assumes that all energy can be accounted for, and an equilibrium can be obtained. It is a view-independent method, because it handles interaction only between light sources and surfaces. However, due to the mentioned assumptions, it only handles diffuse-to-diffuse reflections of light.

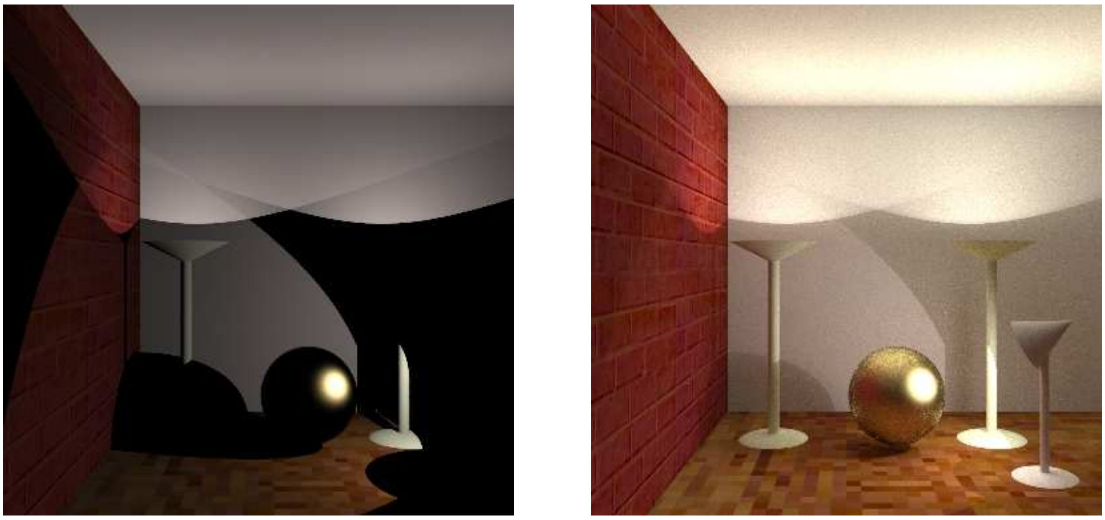
The radiosity algorithm works in object space, and in coarse steps, it does the following: mesh surfaces into elements, compute their inter-reflections, solve linear system for radiosities and lastly reconstruct scene and display it.

Ray tracing Ray tracing works with three “ideas” about light: its rays travel in straight lines, rays do not interfere if they cross and light travels from light source to eye, but due to reciprocity, the physics are invariant if the path is reversed. Ray tracing therefore traces rays from the eye into the scene through all pixels in the image plane. All rays are tested against all geometry. It handles transparency, shadows, specular inter-reflections and texturing in a straight-forward manner. It is said to be view-dependent, because it is limited to the current view and image plane. However, the radiance within the scene of course remains the same no matter where its viewed from.

Because of the relatively course resolution when only sampling once per pixel, ray tracing is prone to aliasing errors, resulting in jagged edges of geometry. However, this is resolved using multiple rays per pixel. Ray tracing is ideal at handling specular-to-specular reflections and can, in its basic form, handle diffuse-to-specular reflection to some extend.

An attempt at solving the rendering equation that extends and combines the concepts of radiosity and ray tracing is presented in [WCG87], where the best of two worlds is combined to form a better solution to realistically simulate light transport, and thus rendering true-to-life complex scenes. It is handled in a two-pass manner: first a view-independent radiosity computation pass, followed by a view-dependent ray tracing pass.

An example of a scene rendered using a global illumination method is shown in Figure E.1(b). In comparison to local illumination methods normally used in real time rendering, exemplified in Figure E.1(a), the global illumination methods handles both shadows and inter-reflection. Strictly speaking though, local illumination is the consideration of light in one point and directly from light sources, without considering any other points or geometry in the scene. This is why it is wrong that the ball casts a shadow in Figure E.1(a).



(a) “Local illumination”. There is a lack of definition in areas not directly affected any light source.

(b) Global illumination. All areas are well-defined, and shadows are handled.

Figure E.1: Figures illustrating the concepts of local and global illumination. Taken from `http://www.cgl.uwaterloo.ca/Projects/rendering/Talks/swc/`.

F Camera Calibration

This appendix analyzes how points in world space are projected onto the image plane. The parameters of the projection can be divided into extrinsic and intrinsic parameters. The extrinsic parameters describe the pose of the camera. The intrinsic parameters describe how points are projected onto the image plane and how they are distorted. The main sources for this appendix are [TV98] and [Bou07]

F.1 Camera Model

Figure F.1 illustrates the utilized camera model. Using this model, the transformation of world point to image points can be described. In the figure, four coordinate systems are depicted. These are used in the projection from world space to pixel space. A known point $\mathbf{p}_w$ in world space is first transformed into camera space. The coordinate of the point is now $\mathbf{p}_c$. Next, the point is projected onto the image plane using a perspective projection resulting in the point $\mathbf{p}_u$ expressed in camera coordinates. If the camera did not have any distortion the final image point is $\mathbf{p}_u$.

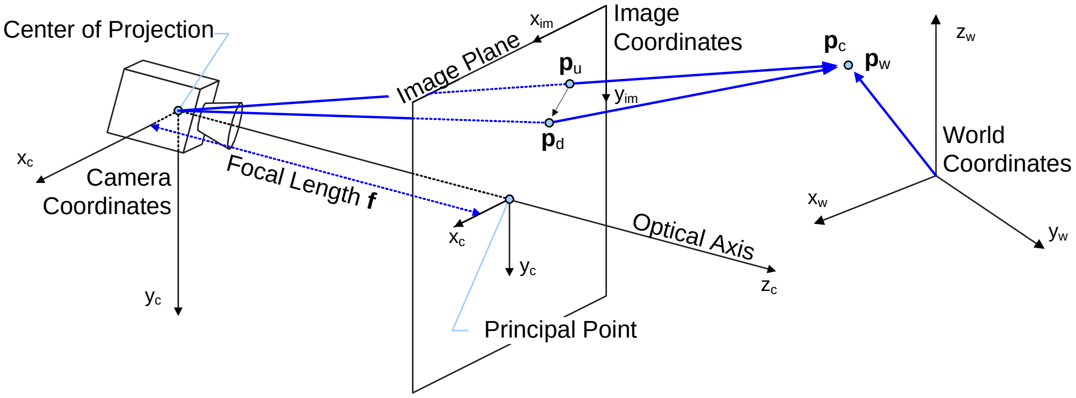


Figure F.1: The coordinate systems used in the projection of a point in world space to image space. The projection of p_w is based on a perspective projection.

However, for real cameras the pinhole model is not an adequate model. The lens introduces distortion which needs to be modeled. Therefore the final distorted point in the image is $\mathbf{p}_d$. The steps involved in transforming the world space point $\mathbf{p}_w$ into image space can be summed up as:

1. A rigid transformation from world to camera space. The transformation involves a rotation and a translation.

2. A perspective projection in camera space of the point onto the image plane using the pinhole model with focal length f.
3. A distortion transformation from $\mathbf{p}_u$ to $\mathbf{p}_d$.
4. A transformation from camera coordinates to image coordinates.

The parameters involved in step 1 are called the extrinsic parameters. The extrinsic parameters are camera independent. The parameters used in steps 2 - 4 are called the intrinsic parameters and are camera dependent.

In this work the MATLAB camera calibration toolkit by [Bou07] is used. Using a checkerboard, the parameters can be estimated from multiple images from different orientations where the corresponding world and image coordinates are known.

F.2 Extrinsic Parameters

The extrinsic parameters transforms a point in world space to camera space. The transformation consists of a rotation followed by a translation. The rotation aligns the axes of the world coordinate system to the camera coordinate system. The translation moves the origin of the world coordinate system to the origin of the camera coordinate system. Knowing the rotation and translation transformations, the camera coordinate of a point in world space can be found as:

$$\mathbf{p}_c = \mathbf{R} \cdot \mathbf{p}_w + \mathbf{t} \tag{F.1}$$

where

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad \text{and} \quad \mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} \tag{F.2}$$

The extrinsic camera parameters are used to determine the camera position relative to the surface. Using this position, the view direction to all texels in the extracted texture can be estimated.

F.3 Intrinsic Parameters

The intrinsic parameters are unique for a camera. However, when estimated they remain the same unless the focus of the camera is changed.

The intrinsic parameters describe how a point is projected, distorted and transformed into image coordinates. The perspective projection can be seen in Figure F.1, where the point $\mathbf{p}_c$ is projected onto the image plane as the undistorted point $\mathbf{p}_u$.

$$\mathbf{p}_u = \begin{bmatrix} f \cdot \frac{x_c}{z_c} \\ f \cdot \frac{y_c}{z_c} \end{bmatrix} \tag{F.3}$$

The distortion is comprised of different types of distortions. Normally only the radial and tangential distortions are considered, but other types exist and can be modeled. Using the radial and tangential distortion, the relation between the distorted and undistorted point is:

$$\mathbf{p}_d = \mathbf{p}_u \cdot \text{radial} + \text{tangential} \tag{F.4}$$

The radial distortion is large for lenses with a large Field of View (FOV); i.e. fisheye lenses. The cameras used in this work have a more narrow FOVs resulting in smaller radial distortion. The radial distortion component is given by:

$$\text{radial} = 1 + k_1 \cdot r^2 + k_2 \cdot r^4 \tag{F.5}$$

where $r^2 = x_u^2 + y_u^2$. The radial distortion is large near the edges of the image and small near the center.

The tangential distortion is due to imperfect centering of the lens components and other lens defects [Bou07]. The tangential distortion is defined as:

$$\text{tangential} = \begin{bmatrix} 2 \cdot k_3 \cdot x_u \cdot y_u + k_4 \cdot (r^2 + 2 \cdot x_u^2) \\ k_3 \cdot (r^2 + 2 \cdot y_u^2) + 2 \cdot k_4 \cdot x_u \cdot y_u \end{bmatrix} \tag{F.6}$$

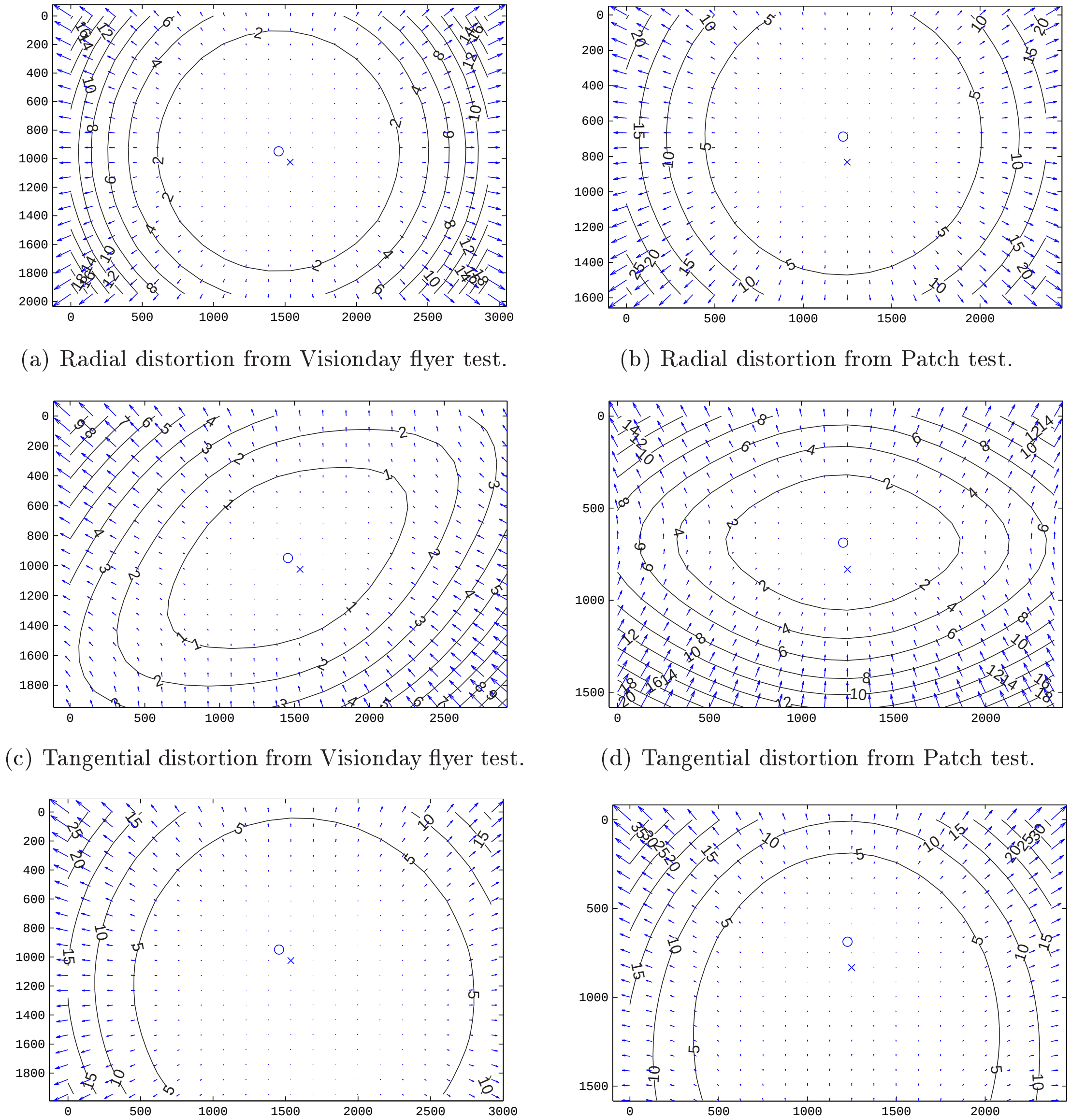
Generally, the tangential distortion is much smaller than the radial. Like the radial distortion it is more visible towards the image edges.

Next, the image coordinates can be found be applying knowledge about the physical dimensions of a pixel (s_x and s_y), which can be found in the most camera data sheets. However, in practice the dimensions are also estimated by the MATLAB implementation. Accounting for the principal point C, the image coordinates can finally be found as:

$$\mathbf{p}_{im} = \begin{bmatrix} \frac{x_d}{s_x} + C_x \\ \frac{y_d}{s_y} + C_y \end{bmatrix} \tag{F.7}$$

Examples of the distortion from the tests performed in Chapter 16 are visualized in Figure F.2. The two plots are different since a different focal length and focus is used in the two tests. The distortion is most dominant towards the corners, where a pixel can be displaced up to 30 pixels. However, the object of interest are most placed near the center of image, where the distortion is low.

When the camera calibration has been accomplished, the image position of a real world point can be calculated. This is useful when sampling known world point on a surface using photographs.



(a) Radial distortion from Visionday flyer test.

(b) Radial distortion from Patch test.

(c) Tangential distortion from Visionday flyer test.

(d) Tangential distortion from Patch test.

(e) Complete distortion from Visionday flyer test.

(f) Complete distortion from Patch test.

Figure F.2: The distortions for Visionday and patch test. The arrows illustrates the displacement caused by the distortion. Visualized using [Bou07].

G Enclosed DVD

G.1 Content of the DVD

The DVD contains the following folders:

/**Thesis**/ The thesis in .pdf, .ps and .tex-format.

/**Application**/ Contains the executable visualization application with the needed .dll-files and parameter maps as well as the source code.

/**Test**/ Contains a modified version of the visualization application where it is possible to see the visualization of the data used in the tests. It also contains figures from all tests.

/**Matlab scripts**/ Contains the developed Matlab scripts for creating observation maps, fitting models to the data therein and plotting it.

/**Data**/ Contains the observation maps for all the tests.

/**Video**/ Contains videos recorded from the application using different parameter maps with different resolutions for both the real surfaces.

G.2 Controlling the Application

Enclosed on the DVD, in the folder /**Application**/ is the working application that can visualize the parameter maps obtained for the two surfaces described in the evaluation. Holding down the left mouse button and moving the mouse can be used to move the surface around. Zooming can be done with the right mouse button. Figure G.1 shows a reference for the use of the numerical keypad for changing between parameter maps, sizes thereof, γ for tonemapping and rendering modes. Besides the numerical keypad and the mouse, the following keys can be used in the application:

z switches between free movement on the hemisphere above the surface, and movement constrained to the view directions used to build the parameter map.
x / c moves to the previous / next of the view directions
v changes between a 1000mm x 1000mm quadrilateral and an A4-sized quadrilateral
p changes texture between “green”, “bricks” and “rocks” textures
e / q decrease / increase tessellation of the quadrilateral
a / d decrease / increase the azimuth
w / s decrease / increase the zenith
r / f zooms in / out
g grab screenshot (filename is screenshot.bmp)
m toggle menu
t toggle tonemapper
ESC quits the application

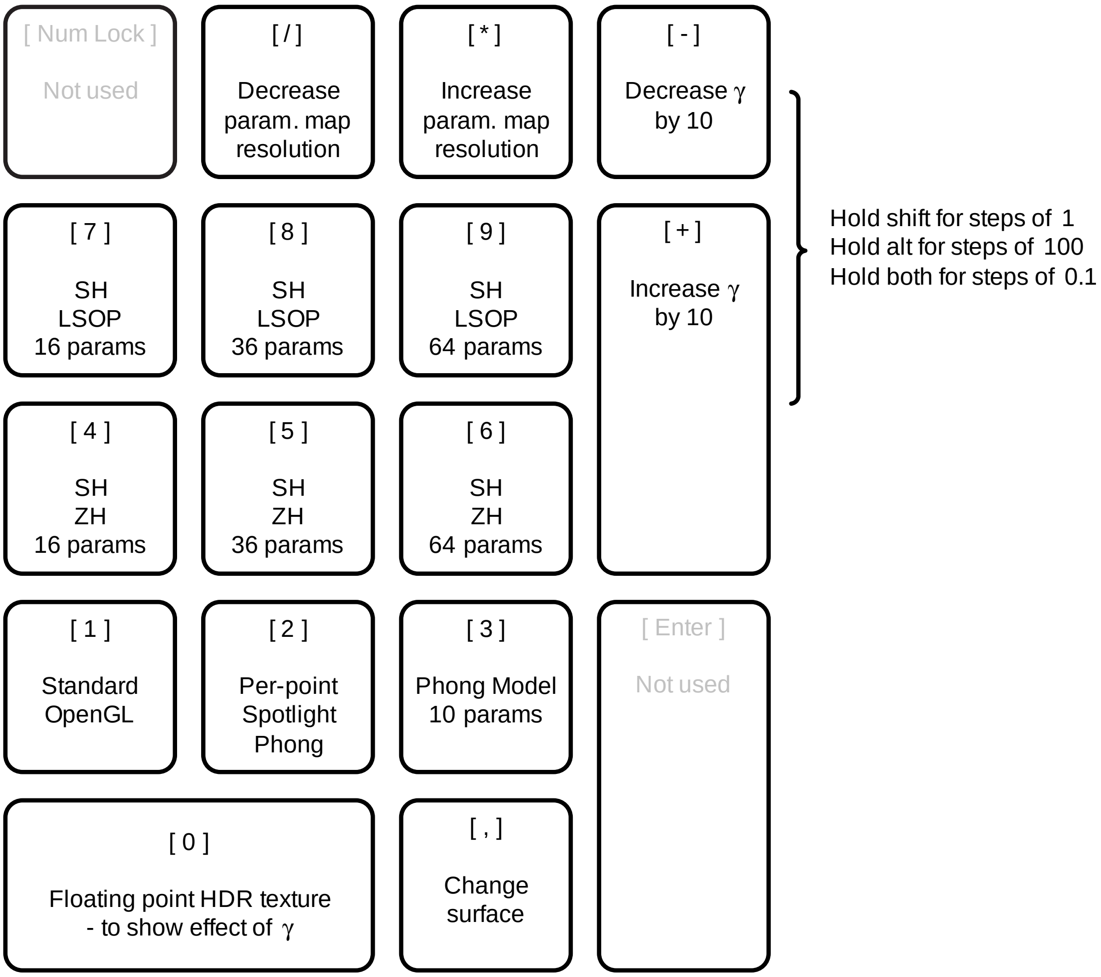


Figure G.1: A reference chart for the numerical keyboard.