Universiteit Utrecht

Master Thesis

Symmetry Improved Photon Maps

ICA-3904296

Author:
Yasen Atanasov

Supervisors:
Dr. Michael Wand
MSc. Marries van de Hoef

September, 2014

Abstract

In recent years there has been a tremendous effort in improving the performance of various global illumination algorithms. Most of this effort has been focused on maximizing the utilization of available CPU and RAM resources, as well as improving the parallel performance of the algorithms. This thesis provides an alternative approach towards improving global illumination by looking at similar areas within the image. A lot of synthetically generated images used in practice exhibit a large amount of repetitive or smooth areas. Combined with the recursive nature of global illumination this suggests that a reuse strategy might, in fact, prove very effective. This thesis proposes a novel approach towards tackling the global illumination problem. We compute an approximate global illumination solution using photon maps and reuse the computed illumination in different areas of the scene to improve the solution. The scene is separated into symmetrical segments and parts of the scene, that share similar radiance are identified. By using the symmetrical relationships between the segments, these areas are then filtered with a version of the non-local means filter that works on photon maps. An analysis of the visual quality of the resulting images as well as a study of convergence behaviour using mean square error is provided. We show that a reuse strategy, based on local geometric symmetry, can be a viable approach toward improving global illumination algorithms as well as provide a stepping stone for further research in the area.

Contents

1 Introduction **2**

2 Related Work **5**
2.1 Rendering and Global Illumination 5
2.1.1 Path Tracing 5
2.1.2 Photon Mapping 6
2.1.3 Radiosity 6
2.2 Local and Partial Geometric Symmetry 7
2.3 Using Symmetry for Rendering 8

3 Symmetry Improved Photon Maps **10**
3.1 Photon Mapping 10
3.1.1 The Photon Map 10
3.1.2 Photon Tracing 10
3.1.3 Radiance Estimation 12
3.1.4 Rendering 13
3.2 Non-Local Filtering 15
3.3 Reusing Lighting Information 16
3.3.1 Prerequisites and Theory 16
3.3.2 Descriptor 17
3.3.3 Searching and Weighing 18
3.3.4 Filtering and Rendering 19

4 Implementation and Experiments **21**
4.1 Implementation 21
4.1.1 Descriptors 21
4.1.2 Searching and Weighing 22
4.1.3 Filtering and Rendering 24
4.2 Results and Discussion 25
4.2.1 Results 26
4.2.2 Discussion 31

5 Conclusion **33**
5.1 Future Work 33

Chapter 1

Introduction

Since the introduction of algorithms such as radiosity [14] and path tracing [19], global illumination has become a very popular, and important, subject in the field of computer graphics. Technically speaking, shadows, reflections, and refractions can also be viewed as examples of global illumination, since in these phenomena one object affects the rendering of another. However, this thesis will be focusing on diffuse inter-reflection - the indirect lighting reflected between different objects. Another example of global illumination effects is caustics - specularly reflected, or refracted light onto another surface. What is most typical about these phenomena is that, they model multiple, recursive interactions, often between a large number of objects, which makes them computationally very expensive. So far, a number of methods have been proposed, that successfully model global illumination. However, generating a high quality, noise-free image in reasonable time still remains a significant challenge.

Illumination Reuse

A tremendous amount of work has been done, in the past decade, to improve computational complexity and utilization of hardware, specifically RAM and CPU power. In recent years there has been a strong inclination towards parallel algorithms, mostly driven by the increase in power from graphical processing units. The most notable examples are ray-tracing and path-tracing, which are particularly well suited for parallelization.

With this recent focus on computational complexity some areas of improvement, like reuse, have been rather unexplored. Since global illumination is recursive by nature, there seem to be large amounts of repetitive calculations that have to be performed in order to compute it. It is obvious that a very big part of the computer generated imagery exhibit some sort of repetition, areas of uniform lighting or gradients, and it is very tempting to try and reuse as much of the information computed for rendering to enhance or speed-up the process.

Correspondingly, approaches that facilitate the reuse of calculations during rendering have started to emerge. Some approaches gather additional data in each pixel, such as distribution of samples or variance, which can then be used to denoise the generated image. We will refer to this sort of methods as *image space*. A good example of an image space method is the adaptive sampling and filtering strategy, proposed by Rousselle et al. in 2012 [23], that takes advantage of the non local filtering method, developed by Buades at al. in 2005 [7]. However, image space methods seem to focus more on the generated samples and their properties, than on light transport within the scene. Alternatively, approaches that reuse parts of the light transport, such as sub-paths, have also been developed. Those will be referred to as *world space* methods. An example of reusing light transport in world space is the method developed by Zhao et al [26]. The authors propose precomputation-based approach by using blocks of

voxels, in which light transfer was computed before rendering. These blocks were then tiled to produce high fidelity cloth renders.

Geometric Symmetry Detection

What is generally understood by symmetry is an invariance under a set of transformations. Two objects are considered symmetrical if there is a (reversible) mapping that transforms one into the other and vice versa. [22]. The field of geometric symmetry detection, which has also had a significant development in recent years, provides a foundation for this thesis. In 2006 Mitra et al. [22] proposed an algorithm for detecting partial and approximate symmetries in geometric models. The method was based on extracting descriptors of local shapes and clustering those to acquire symmetrical correspondences within the model. In 2009 Bokeloh et al. developed an approach based on extracting feature lines to detect parts of a three dimensional scene that were symmetrical under rigid motion. The approach worked very well with large amounts of scanned data.

Much like the geometrical models that comprise it, a rendered scene exhibits a lot of repetition, continuous symmetries and gradients. All these elements suggest that a reuse strategy for lighting, based on geometrical similarity can give promising results.

Objective

The goal of this thesis is to study the connection between geometric symmetry detection and global illumination. More precisely, can local geometric symmetry be used to improve the computation of diffuse interreflection? As an answer to this question a method will be proposed, which provides and improvement on direct density estimation rendering, by non-locally filtering photon maps in world space.

The main idea of the method is to decompose a scene into geometry that is symmetrical under rigid motion and use that underlying decomposition to find areas where the incoming illumination is also symmetrical under rigid motion. That is, under a certain transformation, the radiance reflected from two elements is equal at every point, or at least up to the quality of the global illumination solution. Thus, by decomposing the scene's geometry into symmetrical pieces, one can ensure geometrical symmetry as a necessary condition for achieving, what can be referred to as *illumination symmetry*. Next step is detecting possible symmetries in the incoming light fields of the symmetrical segments - finding symmetrical pieces that would be similarly lit. In order to avoid the perils of high-dimensional representations, a uniform diffuse material across the scene is assumed, which allows us to work with flux in the confines of the photon mapping technique. The approximate solution, generated by photon mapping, can be used to make predictions about the lighting in the scene and detect segments which might share similar incoming light in a sense create a map of the illumination in the scene.

Contribution

The main contributions are as follows:

- A basic theoretical model for reusing lighting based upon he similarity of the underlying geometry. More precisely, a set of predefined tiles will be used, similar to radiosity [9], that are symmetrical under rigid motion and compare their incident light flux;
- A descriptor that holds the information about the incident flux on a given tile and will allow for comparison of the incoming light from different tiles.

- A filtering strategy, inspired by [23] and [7], that implements the idea of non local filtering in world space.
- Visualization method for rendering the filtered photon map. Combining patches and rendering the filtered photon map is a non-trivial task and it has to be approached carefully to obtain the best results.

Overview

The thesis is organized in the following manner:

Related work is described in Chapter 2, first the main classes of Global Illumination techniques are described. Subsequently, the latest in geometric symmetry analyisis and inverse procedural modelling is described. The chapter finishes with a detail overview of other existing approaches that utilize lighting reuse.

In Chapter 3 the baseline algorithm of Photon Mapping is discussed in detail, also other important work like the non local means filtering method. In the second part of the chapter the algorithm is proposed and explained in detail, along with the parameters on which it depends.

Chapter 4 describes the implementation specific of our algorithm, as well optimal values for the parameters. The second part of the chapter provides results and comparative visual as well as numerical analysis of the algorithm.

In Chapter 5 will give a conclusion, as well as suggest improvements as future work.

Chapter 2

Related Work

This chapter will provide a brief overview of related techniques and ones that were influential to the approach, described in later chapters. Section 2.1 contains an brief overview of several approaches for computing global illumination. The section itself is split into three parts - Path Tracing, Photon Mapping and Radiosity. Section 2.2 describes work in the area of geometric symmetry analysis, while Section gives a detailed overview of the current approaches to reusing lighting calculations.

2.1 Rendering and Global Illumination

We will start with an overview of the main classes of rendering techniques for global illumination.

2.1.1 Path Tracing

The basic ray-casting algorithm has been around since the 60s. The approach was first described by Appel in his paper on visualisation of three dimensional shapes.[1] However, it wasn't until the 80s when improvements of computing machinery allowed scientist to use this approach in more sophisticated rendering techniques.

An important step towards computing global illumination is Distributed Ray Tracing. Proposed by Cook in 1984, this is a technique that treats the illumination problem as a set of nested integrals of an n-dimensional function and distributes samples along each dimension.[10] For instance, shooting rays at different times, to achieve motion blur, or distributing reflected rays around the mirror direction to achieve glossy reflections. The algorithm works with very little overhead when combined with anti-aliasing - by using the additional super-sampling rays.

In 1986 Jim Kajiya proposed approaching the global illumination problem as a recursive intergral equation, in which the incident light at each point is dependant upon the radiated light from every other visible point in the scene - *The Rendering Equation* .

$$L_o(x, \vec{\omega_o}) = L_e(x, \vec{\omega_0}) + \int_\Omega f(x, \vec{\omega_i}, \vec{\omega_o}) L(x', \vec{\omega_i}) g(x, x^1) d\vec{\omega_i}$$

The author also provides a straightforward Monte Carlo method for numerically solving this multidimensional equation called *path tracing*. The core idea behind path tracing is to sample one continuous path through the scene, via ray-tracing, at each point choosing a random direction based on the geometry. The paths start from the camera and are averaged continuously to achieve an solution that converges to the expected value. [19]

There have been a number of improvements to the path tracing strategy. Most often a target for improvement is the convergence speed of the algorithm. Especially in complicated scenes, where paths to the light source have low probability because of occlusion or other factors.

The main idea behind Bi-Directional Path Tracing is to shoot paths, both from the camera as well as the eye. In a second pass these rays are connected using shadow rays. [21]

In 1997 Veach et al. [24] introduced Metropolis Light Transport, an approach that improves calculation of global illumination in difficult scenes by using a *Markov Chain* Monte Carlo sampling method. The method is based on a mutation strategy of a *current* light transport path. Each time a mutation is executed the new path is evaluated and rejected (or accepted) with a certain probability, based on the actual contribution that the path would have on the ideal image. Since its sampling strategy allows for a more thorough exploration of difficult to find paths, the approach preforms particularly well in difficult scenarios, such as bright indirect light, or light incoming from holes in the geometry.

2.1.2 Photon Mapping

With all the mentioned improvements, Monte Carlo path tracing has its downsides. Particularly, the amount of noise and slow convergence when dealing with refracted light.

In 1986 James Arvo proposed Backward Path Tracing [2] as a an extension to path tracing. The method successfully handles diffuse interreflection and caustics. The main idea of the method is to split the rendering into two passes. In the first pass rays are emitted from the light sources and are recursively bounced around the scene, depositing some of their energy at each bounce. Each diffuse surface is enhanced with a texture map that accumulates the incident light at each corresponding pixel. These texture maps are referred to as *illumination maps*. The recorded illumination is then used in a second pass, when rays are traced from the eye. At each ray-surface intersection, the accumulated energy from the first pass is divided by the world-space area that corresponds to it and yields the illumination at that point. The algorithm is very well suited to for rendering refraction effects like caustics.

The Photon Map was introduced by Jensen and Christensen in 1994, and in essence, it represents a 3-dimensional kd tree that is used for storing photons. The photons themselves, are emitted from the light sources during a backward path tracing tracing pass. Initially the Photon Map was introduced to handle non-parametrizable objects such as fractals or iso-surfaces - illumination maps were used to handle meshes .[18] However, in later work the Photon map became the sole structure for storing the propagated light. The final image is rendered by splitting the illumination into separate contributions - direct illumination, direct specular, indirect diffuse and specular, etc. The indirect diffuse illumination and caustics are rendered using the photon maps by estimating photon density.[17]

Progressive Photon Mapping extends the original approach, by allowing for very large amounts of photons to be used in a scene without allocating huge amounts of memory. The method uses a iterative approach, where at each pass a certain amount of photons is shot in the scene and the density estimates at each pixel are updated accordingly. The authors show, that the solution remains consistent throughout the method.[15]

2.1.3 Radiosity

Radiosity class algorithms are an alternative approach to computing global illumination and have some advantages over standard Monte Carlo methods - they lack noise, converge considerably quicker and are view independent. However, only ideal diffuse reflectors are taken into account. Radiosity is based on methods from thermal engineering and makes use of the

finite element method.[14] The basic idea is to split the scene into rectangular patches so that the global illumination in the scene can be represented as a system of equations.

$$\begin{bmatrix} 1-p_1F_{11} & -p_1F_{12} & \cdots & -p_1F_{1n} \\ 1-p_2F_{21} & -p_2F_{22} & \cdots & -p_2F_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 1-p_nF_{n1} & -p_nF_{n2} & \cdots & -p_nF_{nn} \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} \times \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{bmatrix}$$

Each equation represents the transfer of light from one patch to the scene. Here p_i represents the reflectivity of the patch, E_i - the amount of emitted energy, B_i - the total rate of energy leaving a surface (its radiosity) and F_{ij} is called a *form factor* The *form factor* is a reciprocal geometric relationship between two patches and represents the amount of light that is transferred between the surfaces. Form factors for patches, which are far away compared to their relative area can be approximated, while ones that are close can be subdivided until they are small enough to approximate as well. Thus an efficient solution to the above system can be obtained. The form factors are precomputed, and during rendering the system is solved using a *Gauss-Siedel* iterative approach.[9]

In later work, the form factors were computed on the fly, in order to minimize memory requirements. Also, a progressive refinement approach was adopted to allow the rendering of interactive, dynamic sequences. The relationships between the patches are reversed and light is shot from each path into the scene, instead of gathering it. At each step of the iterative solution the difference between the previous estimate and the current one is shot from each patch. By always adding the largest contribution first, a good approximation can be achieved after only a small number of iterations. [8]

Additionally, radiosity can take advantage of current graphics hardware quite successfully. [11] As we will show later, the separation of the scene into square patches is something that also appears in the approach proposed in this thesis.

2.2 Local and Partial Geometric Symmetry

When talking about local symmetry detection in 3D, one must mention the field of texture synthesis. The goal of texture synthesis is taking a texture sample and synthesizing a new image that, when perceived by a human observer, appears to be generated by the same underlying stochastic process.[25] A good example is the work of Efros et al. [13] - a non-parametric method for texture synthesis. The input texture sample is treated as a Markov Random Field, assuming complete independence between the brightness of a pixel, given its neighbourhood, and the brightness of the remaining image. The output image is grown pixel by pixel, from a 3×3 random seed from the initial texture. For each pixel a NxN neighbourhood around it is taken and compared to equally sized patches from the original image and the value for the new pixel is probabilistically computed from windows' centre pixels.

Mitra et al. [22] proposed a cascading algorithm, in order to minimize costly comparisons when detecting partial and approximate symmetries. In the first stage, sampled points from the model are paired under various transformations and populate a so-called *transformation space*. In the second stage, clusters of points are extracted from this space. These clusters represent potentially symmetrical regions and each point represent a vote in favour of a certain transformation. The parts of the model that correspond to the clusters, which have received the most votes, are then verified by directly comparing geometries.

Bokeloh et al. [4] proposed a memory efficient method for detecting rigid symmetries in very large datasets. The algorithm extracts line features using slippage analysis and pairs

these features to create local coordinate systems. Next, the algorithm searches for clusters of such line features based on their local coordinate systems.

One can find the ideas of comparing pixel patches in the r-similarity described by Bokeloh et al [5]. The goal of the authors is to define a *shape-grammar* for synthesizing 3-dimensional models by splitting a given model into r-sized patches (r-neighbourhoods) along boundaries that separate non-symmetric parts of the model. By using context free grammars r-neighbourhoods are then composed into new models, which are r-similar to the original model and to each other. That is, for every point in the new model, the point and its r-neighbourhood can be found somewhere on the old one.[5]

In recent work, the notion of r-symmetry and r-similarity was extended to split a whole 3-dimensional model into a set of equivalence classes of r-sized building blocks.[20]

2.3 Using Symmetry for Rendering

Substantial geometric redundancy in many natural, as well as synthetic scenes, can make reuse a successful strategy. One such example is the work by Zhao et al[26]. The authors tackle the problem of rendering highly complex volumetric cloth models. The key challenge is long paths, with multiple scattering events, especially when rendering brightly coloured materials. This makes rendering, using conventional Monte Carlo path tracing extremely long. The authors propose taking advantage of the repetitive structure of the cloth models, which are made up of relatively small amount of exemplar blocks (tens to hundreds), compared to number of all blocks (millions). However, precomputing illumination is a challenge on its own, mainly because it requires dealing with highly dimensional light fields, which in turn require high-dimensional data representations, especially in a anisotropic medium, such as cloth. The key insight is split the long paths with so-called 'diffuser events' at the boundaries of blocks. Thus the problem becomes about precomputing flux, instead of light fields. Each exemplar block is split into voxels, independent of the model's structure. The out-facing faces of the voxels (or *patches*) are used to handle light transfer between blocks (and insert diffuser events). By tracing paths within each exemplar block, light transfer matrices are computed. Each element of such matrix handles the amount of light transferred from one voxel(patch) to another. After computing the light arriving at each voxel from the light source, light propagation within the volume becomes a question of matrix multiplication. Finally a final gather pass is executed to collect the computed light from each voxel. [26]

An alternative approach is to reuse the light gathered by the pixels in the intermediate stages of rendering. Image space reuse method have become quite popular recently, before we cover those we should mention the non-local image denoising technique described by Buades [7], as seems to have a strong influence on these approaches. As opposed to the usual local smoothing filters, like the gaussian for example, the NL-means makes use of pixels from different areas of an image instead of just the local neighbourhood. The value of each pixel is computed as a weighted average of all the pixels in the image, based on the visual similarity of their surrounding r-neighbourhoods. Thus the weight for each pixel is based not only on the values of the pixels but the geometric configuration around them. We will cover the NL-means filter in more detain in Chapter 3.

The approach described by Reusselle et al. [23] is a good example of a method that makes use of the non-local image denoising technique described above. The method uses the NL-means filter to drive an adaptive sampling strategy for path tracing, in order to produce less noisy images. At each iteration the samples are split between two buffers. Each buffer is then cross-filtered by using weight computed using the other one, in order to ensure that the weights are independent from the input signal. Error estimation is the computed by the

formula

$$E = \frac{(\hat{A}-\hat{B})^2}{\epsilon-\hat{A}^2}$$

Where $\hat{A}$ and $\hat{B}$ are the two filtered buffers, and ϵ ensures that there is no division by zero.

The computed error is then used to drive subsequent sampling - by assigning more samples to pixels with higher estimated error. The authors also make use of additional data from the sampling, like sample count and empirical variance. The empirical variance is then used in distance measurement instead of the traditionally assumed by the NL-means method uniform Gaussian noise. Additionally, the authors propose a new method for calculating the distance to patches on gradients.

Recently, Delbracio et al. [12] proposed an image space filtering technique that takes advantage of the individual samples, computed during rendering, instead of just the final colour of the pixel. In their method, 'Ray Histogram Fusion', the authors propose keeping track of the colour computed by each path, shot from a certain pixel, in 3-dimensional histogram.

The image is then filtered as in [23], the main difference being that complete patches are averaged, not only their centre pixels. By averaging patches, more estimates for each pixel are obtained - since each patch contains $(2\omega+1)^2$ pixels, then each pixel participates in $(2\omega+1)^2$ patches. The colour of each pixel is stored as a 3-dimensional histogram, and the distance between two pixels is measured as a statistical distance of their respective histograms. The main assumption is that pixels that have similar histograms share a similar sample distribution, thus their ideal values should be close.

New values are computed by combining the histograms of similar pixels, and the final colour of a pixel is the average of all the samples shot from all similar pixels. Finally, the authors propose a multi-scale extension of their algorithm to handle low frequency noise. The original image is down-sampled using a Gaussian filter, and the histograms of neighbouring pixels are fused to produce a lower scale image. The process is continues several times, and the denoising algorithm is applied to each scale, which are then combined to achieve the final image.[12]

Chapter 3

Symmetry Improved Photon Maps

This section gives a overview of our approach for reusing lighting information. In order to give some context, the methods our approach is based on will be reviewed in detail first. Section 3.1 gives a thorough explanation of Photon Mapping, while Section 3.2 gives a detailed overview of the NL-means denoising approach. Section 3.3 describes a basic theoretical framework to guide our study as well as a practical approach to reusing lighting.

3.1 Photon Mapping

Photon Mapping represents a two pass approach to rendering global illumination. In the first pass packets of light, or photons, are emitted from the light sources and distributed in the scene via a standard Monte Carlo approach. In the second pass, the emitted light is gathered by shooting rays from the camera, similar to conventional ray-tracing. [17]

The key feature of this approach is, as the name suggests, the Photon Map - a data structure that holds the photons from the initial pass and allows for quick retrieval. The standard method makes use of two photon maps - one low-resolution map for diffuse lighting and one high-resolution for caustics. The use of two separate maps allows to optimize memory requirements and in particular - speed, compared to previous work - the caustic map takes up a relatively small portion of the screen, while the diffuse map is rendered faster without the clutter of additional caustic photons.

3.1.1 The Photon Map

The photon map is modelled as a balanced Kd-tree [17]. The multidimensional binary search tree, more commonly known as kd-tree, is a space partitioning data structure for organizing points in a k-dimensional space. Every leaf node of the tree represents a point in space, the position of which along a certain axis defines a hyper-plane, perpendicular to that axis, that splits space in two. The left subree represents points that belong to the left half-space and the same for the right subtree. The axes, along which space is divided are cycled through as one goes down the tree, so that no two consecutive nodes split space along the same axis.

Searching for a point in the kd-tree nearest to a given location takes at most $O(logN)$ time, where N is the number of points in the tree. [3] In reality the use of the photon map is much more efficient, since the required photons reside in the same parts of the scene. [17]

3.1.2 Photon Tracing

The photons, stored in the Photon Map, represents small packets of energy that are propagated through the scene. Their main attributes are direction, colour and intensity. The

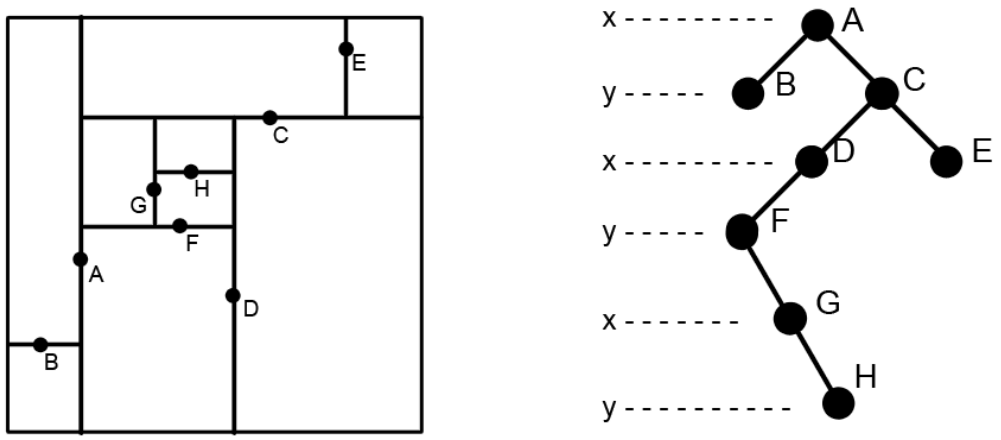


Figure 3.1: An example of a 2-dimensional kd-tree. Left - a set of points, along with the space partition that they define. Right - hierarchical representation of the points as a binary kd-tree, with the axis along which the split has occurred.

approach makes no restrictions about the type, or shape, of the light source the photons are emitted from. The only important condition is that the emitted photons' distribution models the distribution of the emissive power of the light source. Each photon carries a fraction of the light source's power equal to $\frac{L}{N}$, where N is the number of photons shot from this light source, and L is the intensity.

In order to optimize photon casting in scenes with sparse geometry, *projection maps* can be utilized. They represent a render of the geometry from the light's perspective - for a point light it is a spherical map, and for a directional light it is a planar projection. Projection maps give an estimate of where geometry might be, and are especially useful for generating the caustic map.

The photons are propagated through the scene using the *backward ray-tracing* technique. [18] The approach is similar to conventional path tracing, with the exception that photons propagate light flux, instead of gathering radiance. Every time a photon interacts with a surface, it can either be reflected specularly, reflected diffusely or be absorbed. A *russian roulette* approach is used to determine the outcome of the interaction, which means the the power of the photon will not be attenuated. Thus a surface with 75 percent reflectivity will reflect only 75 percent of the incoming photons. The colour of the reflected photon, however, needs to be adjusted based on the colour of the surface. Let x_i be uniformly distributed random variable, and $x_i \in [0, 1]$. The probabilities are computed in the following manner:

$$P_d = \frac{max(d_r P_r, d_g P_g, d_b P_b)}{max(P_r, P_g, P_b)} \tag{3.1}$$

$$P_s = \frac{max(s_r P_r, s_g P_g, s_b P_b)}{max(P_r, P_g, P_b)} \tag{3.2}$$

Here (P_r, P_g, P_b) is the colour of the incident photon and (d_r, d_g, d_b) and (s_r, s_g, s_b) denote the specular and diffuse coefficients of the surface $((d_r + s_r, d_g + s_g, d_b + s_b) <= (1, 1, 1))$. The propagation conditions are as follows:

$$\begin{aligned} &x_i \in [0, P_d) \rightarrow \text{diffuse reflection} \\ &x_i \in [P_d, P_d + P_s] \rightarrow \text{specular reflection} \\ &x_i \in (P_s + P_d, 1] \rightarrow \text{absorption} \end{aligned} \tag{3.3}$$

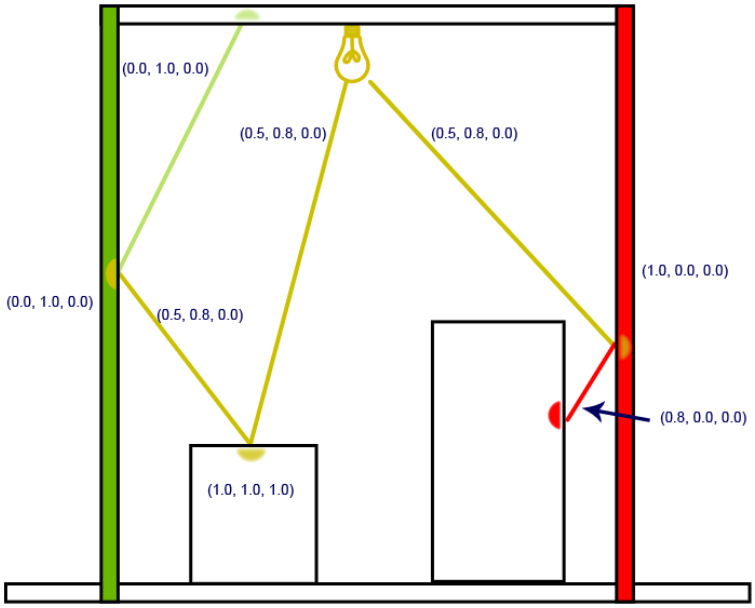


Figure 3.2: Photon tracing within a 2-dimensional scene. The values in brackets represent colour values of the objects in the scene and values of the photons before and after interaction.

In case of reflection the colour of the incident photon needs to be adjusted by using the following formula:

$$\begin{aligned} P_{refl,r} &= P_{inc,r} d_r / P_d \\ P_{refl,g} &= P_{inc,g} d_g / P_d \\ P_{refl,b} &= P_{inc,b} d_g / P_d \end{aligned} \tag{3.4}$$

Where $P_{refl,\{r/g/b\}}$ is the power of the reflected photon, and $P_{inc,\{r/g/b\}}$ is the power of the incident one.

Whenever a photon is reflected diffusely, the interaction is stored within one of the photon maps, along with its position, incident direction and colour. The global photon map represents an approximation of the global illumination solution and holds all photons, that have been reflected diffusely. The caustic map is generated in a separate pass and holds only the photons that have passed through at least one specular reflection. A photon can be stored multiple times during its propagation through the scene. [16]

3.1.3 Radiance Estimation

The usual way of computing the radiance at a given point, formulated by Kajiya is:

$$L_o(x, \vec{\omega_o}) = L_e(x, \vec{\omega_0}) + \int_\Omega f(x, \vec{\omega_i}, \vec{\omega_o}) L(x', \vec{\omega_i}) g(x, x') d\vec{\omega_i} \tag{3.5}$$

Where L_o is the reflected radiance at point x in direction ω, Ω is the unit hemisphere around x, L is the incoming radiance at point x from from point x', n is the normal at x and f is the *bidirectional reflectance distribution function* [19]. However, since the photon map stores flux and not radiance, we need the relationship between the two:

$$L = \frac{d^2\Phi}{dAd\Omega cos\theta} \tag{3.6}$$

Each photon caries a certain amount of energy, so we can estimate the flux at a given point that point as the sum of all the energy carried by incident photons, divided by the area in which they are located. In practical terms, this means expanding a sphere centred at the point x until it contains enough photons, and assume that each photon in the sphere intersects the surface at the point x. We assume the surface being sampled is locally flat, which means means we can project the sphere onto the surface. Thus we obtain the following formula for the radiance estimate:

$$L(x,\omega) = \sum_{p=1}^{n} f_r(x,\omega_p,\omega)\frac{\Delta\Phi_i(x,\omega_p)}{\Delta A} \tag{3.7}$$

Where $\Phi(\omega_p)$ is the power of the photon p arriving from direction ω [16]. In practice, using a sphere to gather nearest photons can cause artefacts near corners - a better solution is using a disk or a ellipsoid. This minimizes most of the artefacts, however not entirely - some problems still remain around sharp corners.[17]

It can be shown, that the radiance estimate is consistent - the quality of the solution increases with the amount of photons. We can set the number of photons in the radiance estimate to N^α, where N is the number of photons in the map, and $\alpha \in (0,1)$. If we then take the limit of the above equation, when N goes to infinity, we end up end up estimating an infinite amount of photons over an infinitesimal sphere. [16]

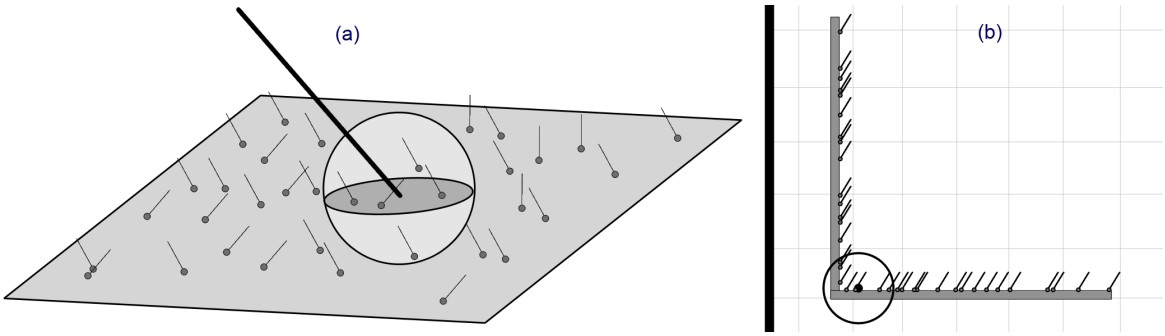


Figure 3.3: Radiance estimation. (a) - Radiance estimation using a sphere on a flat surface, (b) - using a sphere around corners and edges can include the wrong photons in the radiance estimate, causing artefacts.

3.1.4 Rendering

Separating indirect illumination in two different maps, that are generated in separate passes allows for very efficient computation of global illumination. The incoming radiance is split into three contributions:

- L_i - direct illumination from the light source
- L_c - indirect illumination via specular reflection - caustics
- L_d - soft indirect illumination that has been reflected diffusely at least once.

Additionally the $BRDF$ has been split, following the separation of the photons maps,into two functions: a diffuse $BRDF$ - f_d and a specular one -f_s. Thus, the integral from equation 3.1.3 can be represented as a sum of several integrals that can be estimated more efficiently on their own:

$$\begin{aligned} L = & \int_\Omega f L_i cos\theta d\omega \\ & + \int_\Omega f_s (L_c + L_d) cos\theta d\omega \\ & + \int_\Omega f_d L_c cos\theta d\omega \\ & + \int_\Omega f_d L_d cos\theta d\omega \end{aligned} \tag{3.8}$$

Each contribution is computed separately, and $f = f_s + f_d$. The global photon map is visualized by using two separate approaches. The parts of the scene that are directly visible through the camera, or through multiple specular reflections, are evaluated with an accurate estimation, as well as points that are close to edges (in order to avoid artefacts). In all other cases an approximate approach is used. [17]

Direct Illumination

The light coming directly from the light source has the most effect on the scene. For the accurate evaluation standard ray-tracing is used - rays are shot of the camera, and at each intersection the light sources are sampled via shadows rays to check for visibility. To speed up the computation of direct lighting and avoid the visibility test altogether, *shadow photons* can be shot from the light sources into areas that are occluded, to tremendously increase the speed with which penumbras are visualized. The approximate solution for this pass is to directly compute the radiance estimate from the global photon map. [17][17]

Specular Relfections

Visualizing the radiance from highly specular of glossy surfaces via the global photon map will require a tremendous amounts of samples, since probability of having a photon on the reflected side of the camera is zero. For efficiency, standard Monte Carlo path tracing is used, coupled with importance sampling. [17]

Caustics

Caustics light, that has been reflected or refracted at least once, and falls onto diffuse surface - similar to focused light by a lens. Computing caustics, by tracing rays from the camera is extremely inefficient. That is why the caustic photon map is always directly visualized through the radiance estimate, even though that requires a very large amount of samples. Some filtering can be used to improve the radiance estimate of the caustic map. The cone filter applies weight to every photon, based on its distance to the estimation point and has been proven to work quite well in dealing with blurry estimates. [17]

Indirect Illumination

Indirect illumination represents light that has went through at least one diffuse bounce. The accurate solution for indirect illumination is once again obtained through ray-tracing. Rays

are shot from the camera and at each intersection, the hemisphere around the intersection point is sampled. Importance sampling, based on the data in the photon map at that point, as well as the BRDF is used to guide the samples. The second bounces are evaluated by using the approximate solution - the radiance is directly estimated from the global photon map. The only exception is when the second sample lands close to the first one (if for instance the is an edge close by), in which case final gathering is once again performed.[17]

3.2 Non-Local Filtering

The Non-Local mean filter, proposed by Buades et al. [7], [6] approaches the question of denoising image from a different perspective than the standard *neighbourhood filters*. Several methods have been proposed, that rely on averaging the immediate neighbourhoods of a pixel. Instead, the NL-means filter takes advantage of the high degree of redundancy present in natural images. Simply put, the authors assume that a certain n-sized window is present in an image multiple times. They compare not only pixels colour values, but also the geometric configuration of the image in the immediate vicinity around the pixels. The value of each pixel is computed as an weighted average of all the pixels in an mage ν:

$$NL[\nu](i) = \sum_{j \in I} \omega(i,j)\nu(j) \tag{3.9}$$

Where the weights ω depend on the similarity of the pixels' neighbourhoods and are such that: $0 \leq \omega(i,j) \leq 1$ and $\sum_j \omega(i,j) = 1$.

The image ν is divided into a system of connected subsets around each pixel, called *neighbourhoods* and denoted with $\mathcal{N}_i$. The restriction of ν to a certain neighbourhood is denoted with $\nu(\mathcal{N}_i)$. The algorithm does not make assumptions about the shape and size of the neighbourhoods, however for simplicity square regions are assumed.

The similarity between two pixels is computed as a Gaussian weighted Euclidean distance between the grayscale intensity levels of the surrounding neighbourhoods:

$$||\nu(\mathcal{N}_i) - \nu(\mathcal{N}_j)||^2_{2,a} \tag{3.10}$$

That is, each element in the neighbourhood is weighted with a Gaussian kernel centred at i, where a denotes the standard deviation. In turn, the weights are computed as:

$$\omega(i,j) = \frac{1}{C(i)} e^{\frac{||\nu(\mathcal{N}_i) - \nu(\mathcal{N}_j)||^2_{2,a}}{h^2}} \tag{3.11}$$

where h^2 can be thought of as a filtering parameter, that controls the decay of the exponential function, and thus - the decay of weights. $C(i)$ is a normalisation factor defined as:

$$C(i) = \sum j e^{\frac{||\nu(\mathcal{N}_i) - \nu(\mathcal{N}_j)||^2_{2,a}}{h^2}} \tag{3.12}$$

In practice, windows of size 3x3 and 5x5, have shown good balance between noise removal and preservation of fine detail and structure. For computational purposes the algorithm is restricted to a *search window*, assuming a spacial locality of the similar areas.[7]

3.3 Reusing Lighting Information

The main goal of this thesis is to explore the possibility of reusing illumination calculations, and provide a method for filtering photon maps in world space. By this we mean representing a certain, locally flat, area in the photon map as an average of other such regions. In this section an overview of the proposed method, using the approaches described above, is provided, along with some theoretical background.

We want to decompose a scene into geometry that is symmetrical under rigid motion and use that underlying decomposition to find areas where the reflected radiance is similar under the same transformation. We make an intuitive assumption that symmetry in reflected radiance depends on symmetry of incoming light and geometry. Photons are then emitted, in order to compute a rough approximation of the scene's illumination. The emitted light at each symmetrical segment is captured thought descriptors, that can be easily compared under rigid motion. The resulting descriptor space is then filtered using the non-local approach described above. Caution must be taken when preforming the filtering, since we are not working with regularly sampled data - like pixels, but with randomly distributed photons. After filtering each segment's flux represents a linear combination of the incident flux of all the similarly lit segments in the scene.

3.3.1 Prerequisites and Theory

To start with, we are going to outline some of the assumption we make and the requirements needed for our algorithm to work properly.

Symmetry

Generally speaking, symmetry represents invariance under a certain transformation, or a whole set of transformations [22]. This thesis, however, focuses on symmetry under rigid motion - rotation, translation, and reflection.

Two objects will be referred to as symmetrical if they are symmetrical under rigid motion:

$$p_1 = T(p_2) : T \in \mathbb{E}^3 \tag{3.13}$$

In fact, all the transformations we refer to here will be rigid motions, unless stated otherwise.

Given a geometric scene $\mathcal{S} \subset \mathbb{R}^3$, we examine a decomposition μ of $\mathcal{S}$ into a set of connected, non-intersecting, subsets (or patches) - $\{p_i\}$ such that $\bigcup_i p_i = \mathcal{S}$. The decomposition is based on the work of Kalojanov et al.[20] - the patches, in turn, are separated into sets of segments, which are symmetrical to each other under rigid motion. Since rigid transform is a relation of equivalence the blocks are separated into equivalence classes of symmetrical elements $\{\tau\}$, where.

$$\tau = \{p | p_i = T(p_j), p_i, p_j \in \mathcal{S}, T \in \mathbb{E}^3\} \tag{3.14}$$

In practice, decomposing the scene into a single class of equivalent elements is sufficient. In fact, in our implementation later on only one type of elements will be considered. Now, that we have achieved our decomposition and we know of the geometric relationships in the scene, we want to derive similar relationships for the illumination. We start by emitting

a certain amount of light into the scene, enough to give us a good approximation of the illumination.

Illumination

In order to find areas of the scene that look similar in the final rendered image, one must find similarities in the reflected radiance of the geometry. In our case - the symmetrical segments . Simply put, we want to find elements that *look the same* under rigid transformation and up to a certain quality of the solution. Throughout this thesis we will assume that all the objects we are considering share a uniform, diffuse bidirectional reflectance distribution function. We will also define an operator that transforms incoming into reflected light from a certain object. The operator $\mathcal{R}$ operates on light fields, and based on the geometry of an object and its $BRDF$ transforms the light field that irradiates the object into a light field that is reflected (radiates) from it. Given p_1 and $p_2 \subset \mathcal{S}$, which are symmetrical under $\mathcal{T} \in \mathbb{E}3$, and an operator $\mathcal{R}$, that takes an incoming light field and an object, and transforms the light field into an outgoing one, based on the geometry of the object and its $BRDF$. Then:

$$Conv(p_1) = Conv(T(p_2)) \Rightarrow \mathcal{R}(L_{in}, p_1) = \mathcal{R}(L_{in}, \mathcal{T}(p_2)) \tag{3.15}$$

Where $Conv(p_1)$ and $Conv(T(p_2))$ denote the convex hull of p_1 and of p_2 transformed by T, respectively.

Thus, to find similarly looking segments, one must look for ones that are similarly lit. One way to achieve this is by analysing the scene from the segment's perspective and precompute visibility or the possible incoming light. Another approach might be to analyse the dependencies between all the pieces that make up the scene, similarly to radiosity [14]. What these to approaches have in common is high dimensionality. The light field of any given segment is a 4-dimensional function, which in turn depends on the light fields of all other segments in the scene.

It soon becomes quite obvious that this kind of precomputation can be very expensive. We need to find some sort of approach that evades the dimensionality curse. Similar to Zhao et al.[26], instead of working with light fields we work with flux. By assuming a uniform, diffuse $BRDF$ across the scene, we can ignore most of the directional information and view each segment and its incoming light separately from the rest of the scene. Additionally, one can easily transform and compare incoming flux. The photon map represents an approximate solution of the illumination in the scene and is achieved through propagating flux, which is ideal for the purpose of this thesis.

Thus, we have transitioned from symmetry of light transport, towards symmetry of the radiance solution. The outgoing radiance will no longer depend on the direction on the incident light, but on the distribution of the flux on the convex hull of the object. The above relationship becomes:

$$\begin{aligned} Conv(p_1) = Conv(T(p_2)) \;\&\; \Phi(Conv(p_1) = T(\Phi(Conv(p_2))) \\ \Rightarrow \mathcal{R}(L_{in}, p_1) = \mathcal{R}(L_{in}, \mathcal{T}(p_2)) \end{aligned} \tag{3.16}$$

This observation, along with the approximate solution from the photon map, can be used to make predictions about the lighting in the scene and detect segments which might share similar incoming light - in a sense create a map of the illumination in the scene.

3.3.2 Descriptor

In order to create such a map, one needs to be able to efficiently compare the light incident to two symmetrical segments. This task needs a correct representation of the incoming flux

that is also computationally inexpensive. Comparing clouds of photons directly is impractical since it can hinder performance severely. Additionally, we may want to take advantage of any possible symmetries within the segments themselves. A simple, but efficient way to represent the flux would be take advantage of the radiance estimation used while rendering the photon map. Additionally, since we are working with are connected subsets of the scene, that represent manifolds - they can be parametrized easily in two dimensions. This allows us to create a simple descriptor, that can represent the incoming illumination correctly and allows for efficient comparison.

The parametrisation is sampled regularly in a two dimensional grid, at each sample the photon density is estimated using the approach described in Section 3.1. Thus, an image of the incident flux at the surface of the segment is created. The parametrization of the surface, can be done in a number of ways - it can be implicit in the case of parametric surfaces or when working with very simple polygons (quads or triangles) or derived through a texture map, when working with complex meshes. The algorithm makes no assumptions about the shape of the segments, but for simplicity we will use planar square patches.

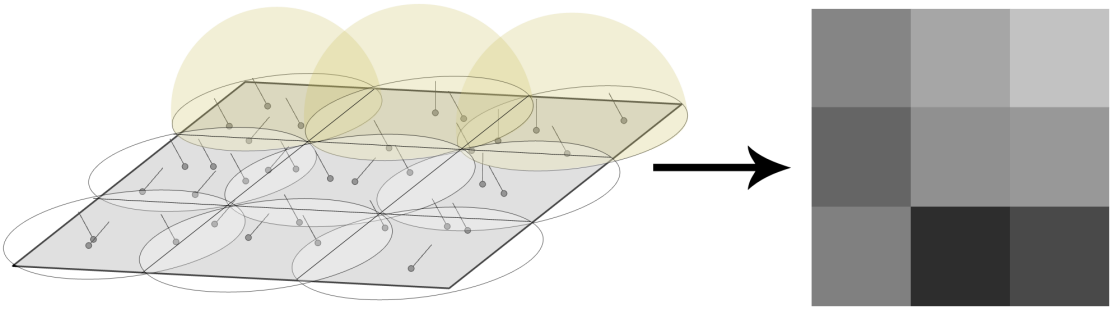


Figure 3.4: A schematic image showing how a descriptor is generated from a geometric segment

Whenever possible, that is the geometry allows it, this image can be rotated or mirrored, allowing for comparison of the descriptors under rigid motion. Sampling resolution and the amount of photons used in density estimation control the balance between quality and performance. In order to represent the flux on the surface of the segment correctly, with respect to the final image, we use the amount of photons, considered in density estimation during rendering.

Less photons would introduce more noise, which in turn will give false negatives. Using more photons would create a blurry estimation, that neglects finer detail that would be captured by the final image, and will create false positives. Through the spacial resolution of the sampling the dimensionality of the descriptor is easily controlled, which allows for an good control over the computational complexity of the filtering algorithm. What is more, the sampling resolution can be made dependant on the amount of photons emitted in the scene, ensuring a balance between performance and quality.

3.3.3 Searching and Weighing

Once a good representation of the illumination in the scene has been obtained, we can proceed with non-locally filtering the photon map. The approach follows the one presented by Buades et al. [7] and described in Section 3.2, however, implementing it in world space required quite a few changes.

Other approaches filter the final image based on information collected from the scene

during rendering, additionally a certain spacial locality is assumed as only samples within a certain window are filtered together [23] [12]. Our approach relies on the idea that geometric symmetry might yield similar lighting in different parts of the scene, thus we ignore the assumption of spacial locality and take all the elements of the scene into account during filtering.

The descriptor described above implicitly defines an n-dimensional space, which houses all the segments of a certain class. We will refer to this space from now on as *descriptor space*. In essence filtering is conducted within the descriptor space and Euclidean distance is used to order the elements based on their similarity to the filtered one. Similarly to the approach described in Section 3.2 all the elements are weighed according to an exponential function.

$$\omega(i,j) = \frac{1}{C(i)} e^{\frac{||\mathcal{D}(p_i)-\mathcal{D}(p_j)||}{\sigma^2}} \tag{3.17}$$

where $\mathcal{D}(p_i)$ is the descriptor of $p_i and\ \sigma^2$ is a filtering parameter that controls weight decay. Similarly $C(i)$ is computed as follows:

$$C(i) = \sum j e^{\frac{||\mathcal{D}(p_i)-\mathcal{D}(p_j)||}{\sigma^2}} \tag{3.18}$$

To ensure good filtering of areas where colour bleeding is present, we must take into account the full colour values at the expense of increasing the dimensionality of the descriptor space. The filtering process can be simply described as applying a n-dimensional Gaussian on the filtered segment and its surrounding elements. Thus, an implicit spacial locality within the descriptor space is assumed - elements further than a certain distance will be practically ignored when filtering. This has additional performance side effects, drastically minimizing the amount of patches used for filtering.

The parameters that affect the quality of filtering are the distribution of elements inside the descriptor space, as well as the weight decay parameter - σ^2.

The quality of the resulting image depends on the amount of photons present in the scene and the amount of photons used in density estimation, as mentioned above. Using a low amount of photons, and will generate a very low frequency representation of the lighting, leading to a more uniform descriptor space, that is prone to more denoising. On the other hand, increasing the number of primary photons, would lead to clustering in the descriptor space and increase in the relative distances within it. This would lead to lesser noise removal in general.

The weight decay parameter, also helps control the filtering effects between *local* and *uniform*, by controlling the number of elements that are taken into account in practice. As in Gaussian filtering, large σ^2 values will create a more blurry, uniform filter, while small values will yield a sharper one.

3.3.4 Filtering and Rendering

The final step in the approach is to filter the similar segments and visualise the new photon map. Since these two steps can be combined they will be reviewed together.

In image space denoising, every pixel is computed as an average of several other pixels. In our case every pixel represents a radiance estimation, at a given point in the scene, which in turn is a function of the incoming flux at and around that point. Thus, the colour of a pixel, displaying the filtered photon map, can be seen as a weighted average of the radiance estimates taken at all the similar patches (possibly all the patches in the scene). Progressive photon mapping [15] is an iterative technique that averages multiple radiance estimation

sample in a consistent way, by modifying the flux and radius of the search sphere. Similar to progressive photon mapping, one can vary the size of the sampling sphere based on the weight of the particular patch in order to keep the averaging consistent. One problem might arise when dealing with samples that are close to edges of segments, and as such gather light from two or more segments, since these segments will most probably have different sets of similar elements with different weights.

Another approach would be to view every resulting patch as weighted average of flux from all the patches in the scene - the resulting patch would hold of the photons from all the similar patches (possibly all the patches in the scene) pre-multiplied by the weight of the corresponding patch. Thus, pixel samples near segment edges will receive a correct distribution of photons, however this approach may lead to additional blur.

Chapter 4

Implementation and Experiments

In this chapter, a detailed explanation will be given, of the implementation of the algorithm outlined in the previous chapter, as well as results. Section 4.1 covers implementation, while Section 4.2 contains results.

4.1 Implementation

Since the focus of this thesis is on computer graphics and finding similarity within the illumination in a scene, geometric symmetry detection was simplified to the point where a set of predefined symmetrical elements were used. The scenes were modelled out of equally sized square segments, that can be subdivided in order to account for the amount of detail in the global illumination.

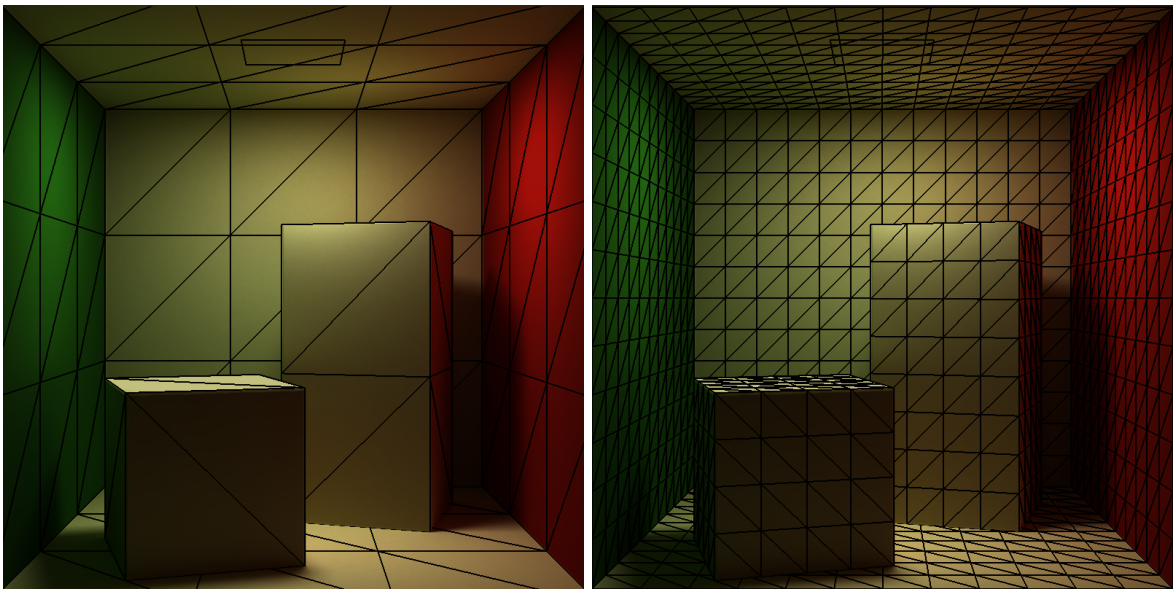


Figure 4.1: The cornell box scene modelled out of identical square elements (left), and subdivided twice(right).

4.1.1 Descriptors

The square segments are very simple to parametrize and are symmetrical w.r.t. rotations of multiples of $\pi/2$ degrees and reflections. The internal symmetry of the segments allows to

compare the descriptors under the corresponding transforms in two dimensions - rotations and reflections, thus potentially increasing the quality of the filtering. The descriptors themselves represent 2-dimensional textures on top of the square segments. The main parameter that controls the behaviour of the descriptors is their resolution.

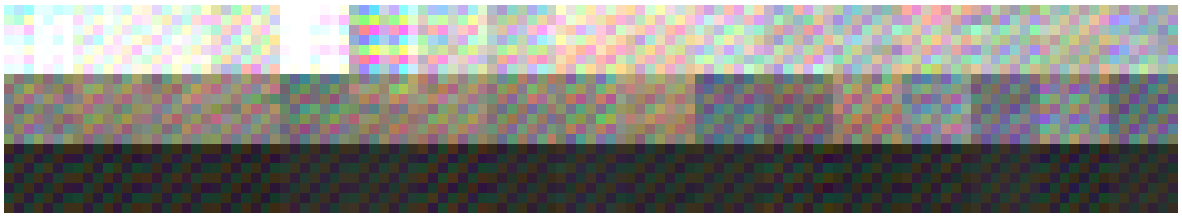


Figure 4.2: Three descriptors along with their nearest neighbours, ordered by similarity. Darker regions seem to contain much more similarity.

The resolution of the descriptors controls how many samples are taken from the surface of each segment, and the dimensionality of the descriptor space. Following Buades et al. [7], [6] relatively low resolutions was used for testing - 3 and 5.

To keep the chosen resolution, while limiting the additional noise in the solution, the segments are automatically subdivided whenever the average amount of photons on each segment exceeds $r^2 \times N_d$, where N_d is the number of photons in the density estimate, and r is the resolution. The idea is to keep the amount of detail captured by each segment relatively consistent with the quality of the final render.

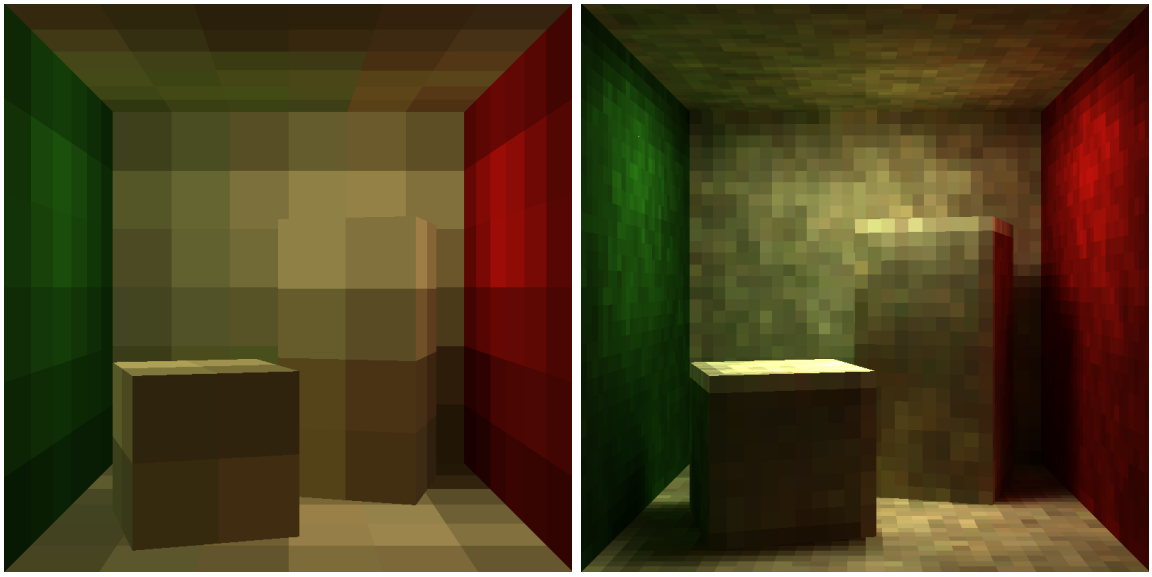


Figure 4.3: The cornell box scene rendered with descriptors applied as textures.

4.1.2 Searching and Weighing

The descriptor space was extended to include the transformed versions of the descriptors - total of eight different versions of each descriptor were used to for filtering. Naturally, this

required some additional bookkeeping, to ensure that there were no duplications and each segment was used only once.

The descriptor space is modelled as a an n-dimensional kd-tree to increase search speed. Two types of searches were performed - a radius search, and a knn search. The radius search takes all the elements within a given sphere into account and weighs them accordingly. The maximum radius can be easily derived from the filtering parameter - σ^2, since after a certain point all weight become zero. However, in practice, using the radius search is very expensive in terms of render times. Since the descriptor space is highly non-uniform, the amount of elements that fall within the search sphere varies from a single one to thousands. A nearest neighbour query has proved to be more practical to use. The amount of elements to use in the knn query can keep filtering times reasonable, while having minimal effect on final result.

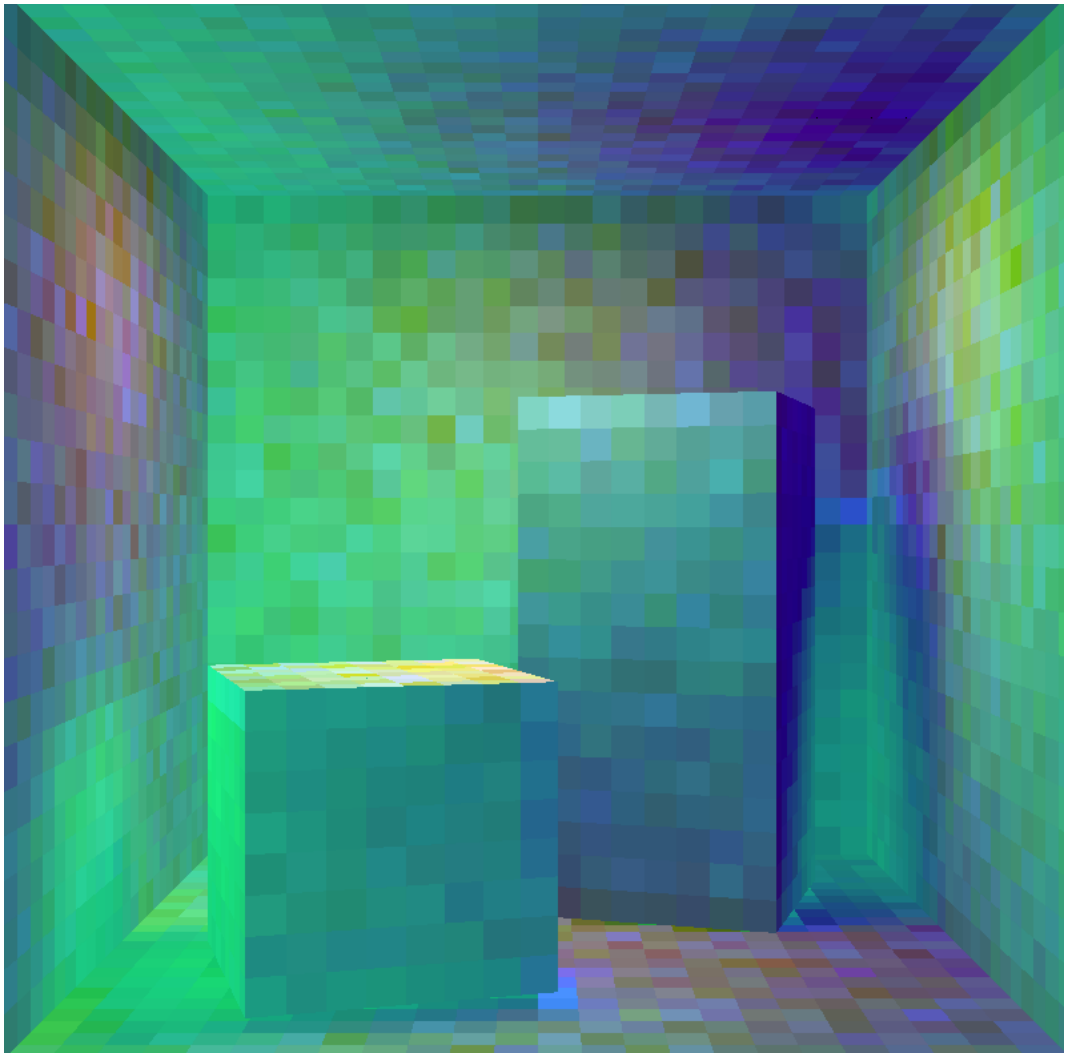


Figure 4.4: The 3-dimensional PCA embedding of the descriptor space rendered as colours of the segments. Similarly coloured elements share a spacial proximity in the descriptor space.

In the sense of image denoising, filtering parameter - σ^2 is set to be dependant on the standard deviation of the noise that is to be removed [6]. While generating a synthetic image, one can theoretically measure the variance present in the image. However, how much of that variance is attributed to noise, and how much is part of the final image is hard to tell. We need a more robust measure of the amount of noise present in the scene. The amount of

noise present in the in photon mapping solution is inversely proportional to $\frac{N_d}{N}$, where N_d is the amount of photons used in the radiance estimate and N is the amount of primary photons in the scene. A low value for this ratio creates noise within the scene, while a high one introduces bias (against which we do not have a solution in this algorithm).

Using k nearest neighbours queries allowed for another approach to selecting the appropriate value for the filtering parameter. A variable σ^2 is chosen every time the descriptor space is queried. In essence an n-dimensional gaussian is fitted onto the resulting neighbours. The standard deviation of the Gaussian is chosen so that the queried element has the highest weight, while the farthest away a weight very close to zero. The results of this approach can be found in Section 4.2.

4.1.3 Filtering and Rendering

Figure 4.5: Iterative rendering of the cornell box scene

For each filtered segments, the new flux is computed by accumulating the flux from all the segments, that participate in the filtering, scaled by the corresponding weight. In order to prevent from using huge amounts of memory the filtering and rendering are done iteratively. At each iteration, a new photon map is constructed. This is done by taking,

for each segment, the closest segment in descriptor space, that has not been used so far, and scaling and transforming its photons correspondingly. The scene is rendered after each iteration and the computed flux is accumulated.

4.2 Results and Discussion

In this section, results and measurements of the algorithm's work will be provided. Experiments were performed to assess the amount of noise removal, and how the algorithm affects the convergence speed of photon mapping. The results are assessed, based on the Mean Square Error, and the amount of removed noise and visual quality. The latter two, hoverer are quite subjective criteria. Measurements of the effect of parameters such as σ^2 is also provided. The section end with a discussion on the method's strong sides, as well as artefacts and weaknesses.

Since, modelling scenes out of square patches is time consuming, we focus our tests on two scenes:

- Cornell Box Scene
- Aligned Cornell Box Scene - a modified version of the above scene, that allows more symmetry in the illumination. Our prediction is that our approach will preform better this sort of scenes.

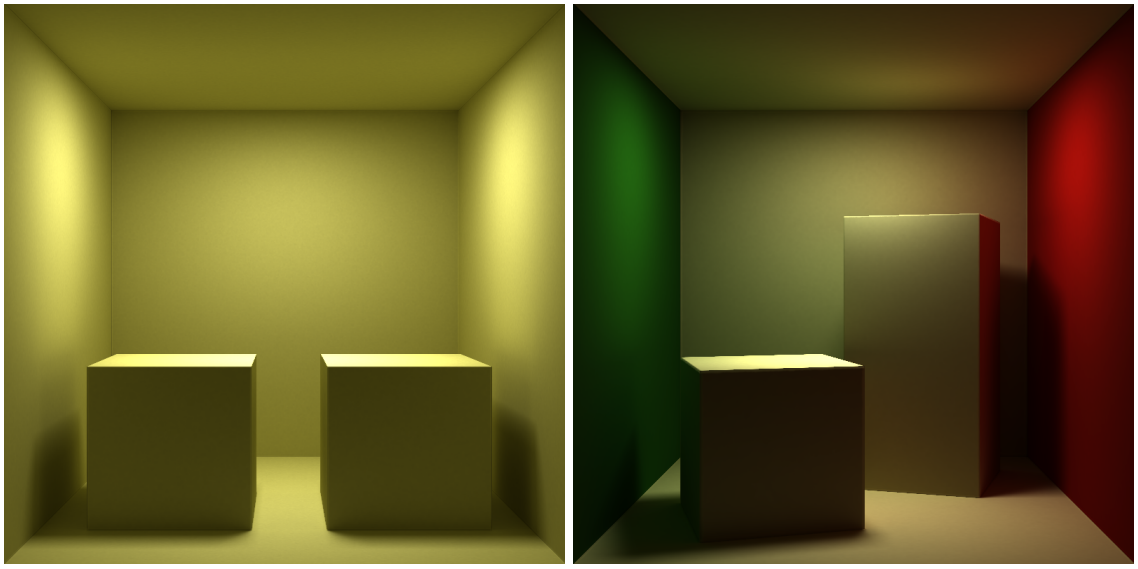


Figure 4.6: Ground truth images of both test scenes.

Theoretical Model

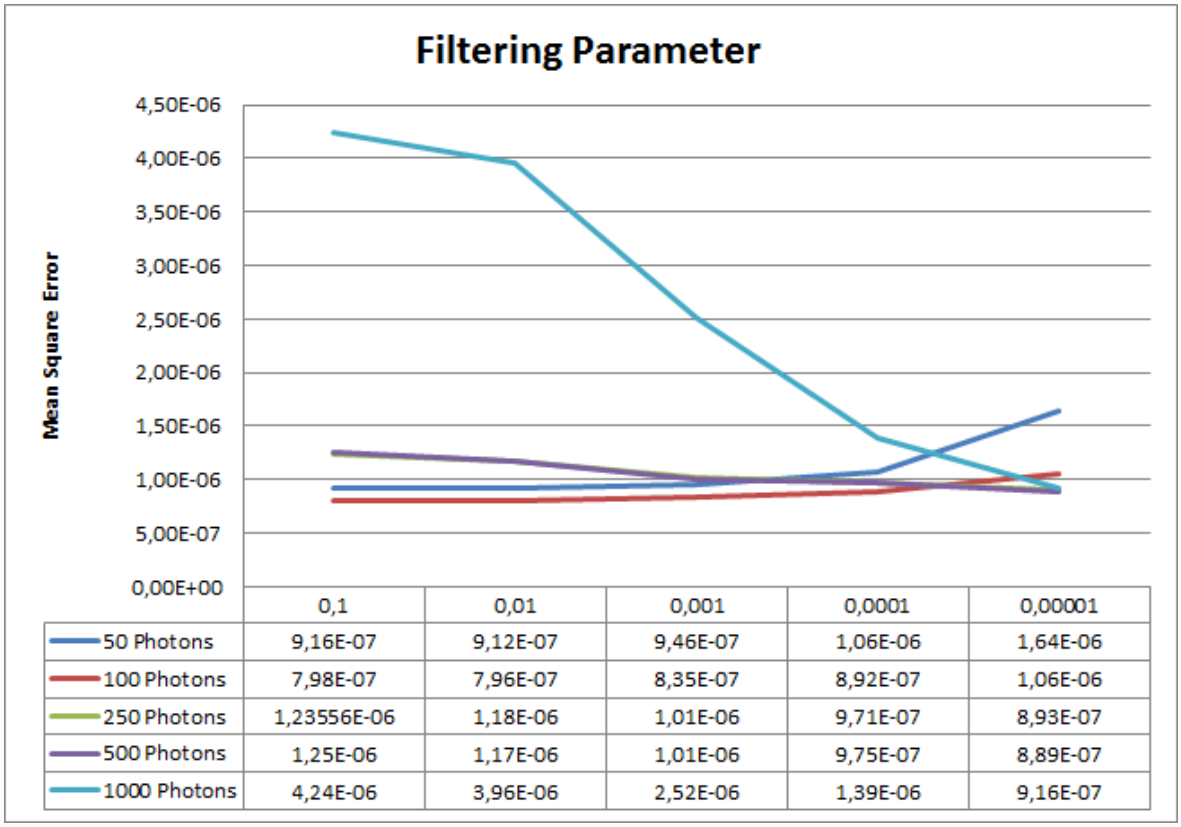
If we assume, as before, that the density estimation photons and the initial photon count are connected through $N_d = N^\alpha$; and $\alpha \in (0, 1)$, and given a fixed resolution of the descriptor. Then, when the number of photons is low and the function is sparsely sampled, the filtering should produce more noise removal (but not necessarily results that are closer to the ground truth). As the amount of samples increase the individual descriptors become more and more distinct from each other, so the contribution of the filtering will be less and less apparent. At the limit (the ground truth) there should be no visible contribution - that is, all the queries will return only one descriptor with weight one, or multiple but very close to each other.

4.2.1 Results

A set of experiments were performed with varying parameters to study how different configurations affect visual quality and convergence speed of the algorithm. For each scene tests were performed to asses how convergence behaviour is affected by changing the resolution of the descriptor, and how the filtering parameter affects quality. The method was tested with different primary photon counts, and resolutions. Throughout all the experiments it is assumed that density estimation photons and primary photons are connected through $N_d = N^{\alpha}$; and $\alpha \in (0, 1)$. Additionally a series of tests were performed by using variable sigma as discussed in the previous section.

Filter Parameter

The filtering parameter, as mentioned earlier, controls the decay rate of the weights. To study the effects of the filtering parameter a series of tests we performed, by using 50, 100, 250, 500 and 1000 photons in the density estimate.



	0,1	0,01	0,001	0,0001	0,00001
50 Photons	9,16E-07	9,12E-07	9,46E-07	1,06E-06	1,64E-06
100 Photons	7,98E-07	7,96E-07	8,35E-07	8,92E-07	1,06E-06
250 Photons	1,23556E-06	1,18E-06	1,01E-06	9,71E-07	8,93E-07
500 Photons	1,25E-06	1,17E-06	1,01E-06	9,75E-07	8,89E-07
1000 Photons	4,24E-06	3,96E-06	2,52E-06	1,39E-06	9,16E-07

Figure 4.7: Mean square error of different configurations of radiance estimate photons and the effect of different σ^2 values on the quality. All test were performed with 1000000 primary photons.

All the test were performed with using 1000000 primary photons and varying values for σ^2. As figure 4.7 shows, as σ^2 decreases, quality seems to slightly decrease for small radii of radiance estimate. However, large radii - using 500 and 1000 photons, show significantly lower quality when tested with high values for σ^2 and the quality improves as the filtering parameter decreases. One would expect that given the scenes used, the bulk of the error would be due to noise on smooth flat surfaces. However, a closer look at the generated images, suggests

that the higher error rates are caused by over-filtering highlights and shadows. As figure 4.8 shows a smooth but over-fingered image, has a much higher MSE than even the noisiest estimates. Another possible reason for this effect is the heavy *blocking* effect, however a careful analysis of figure 4.9 suggests that the bulk of the error in the smooth image is due to missing highlights and shadows and not so much *blocking* effects.

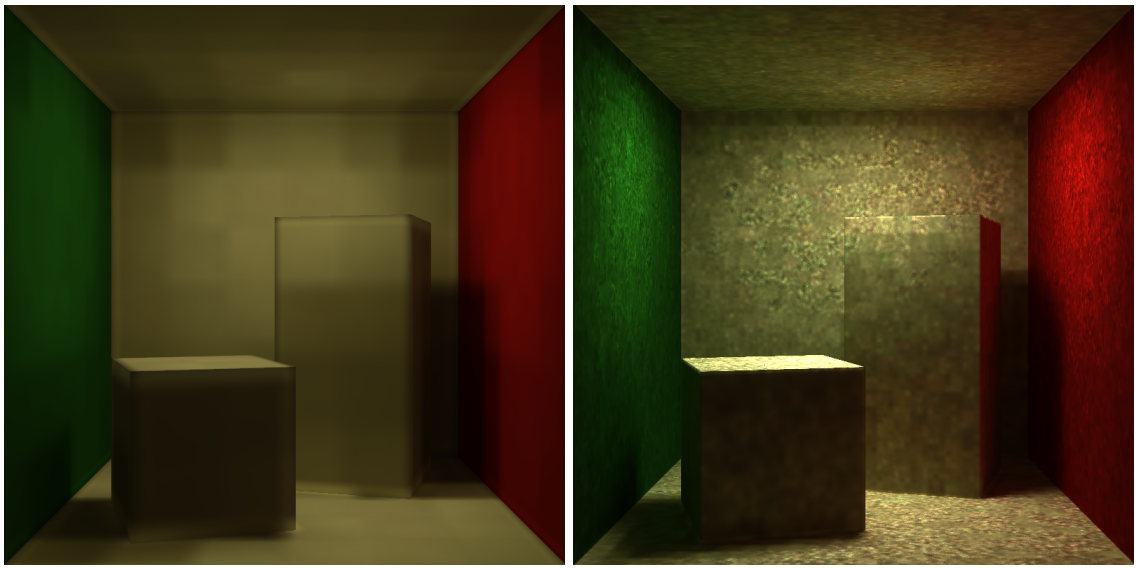


Figure 4.8: A visual comparison of two resulting images from the above experiment. Albeit smooth, the left image has a MSE of 4.23752e-006, while the right one - 1.63928e-006

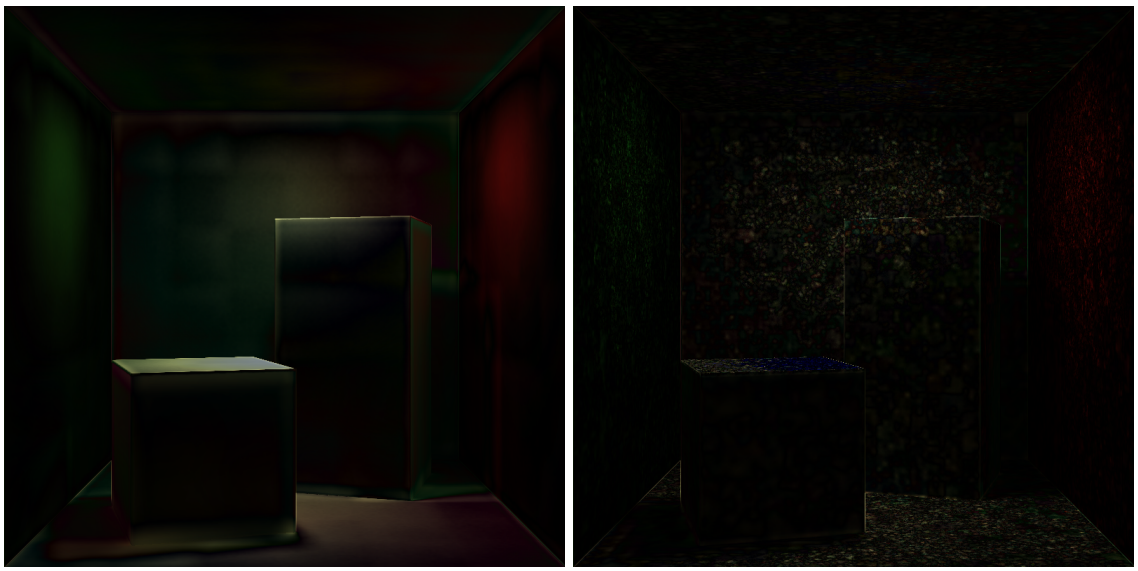
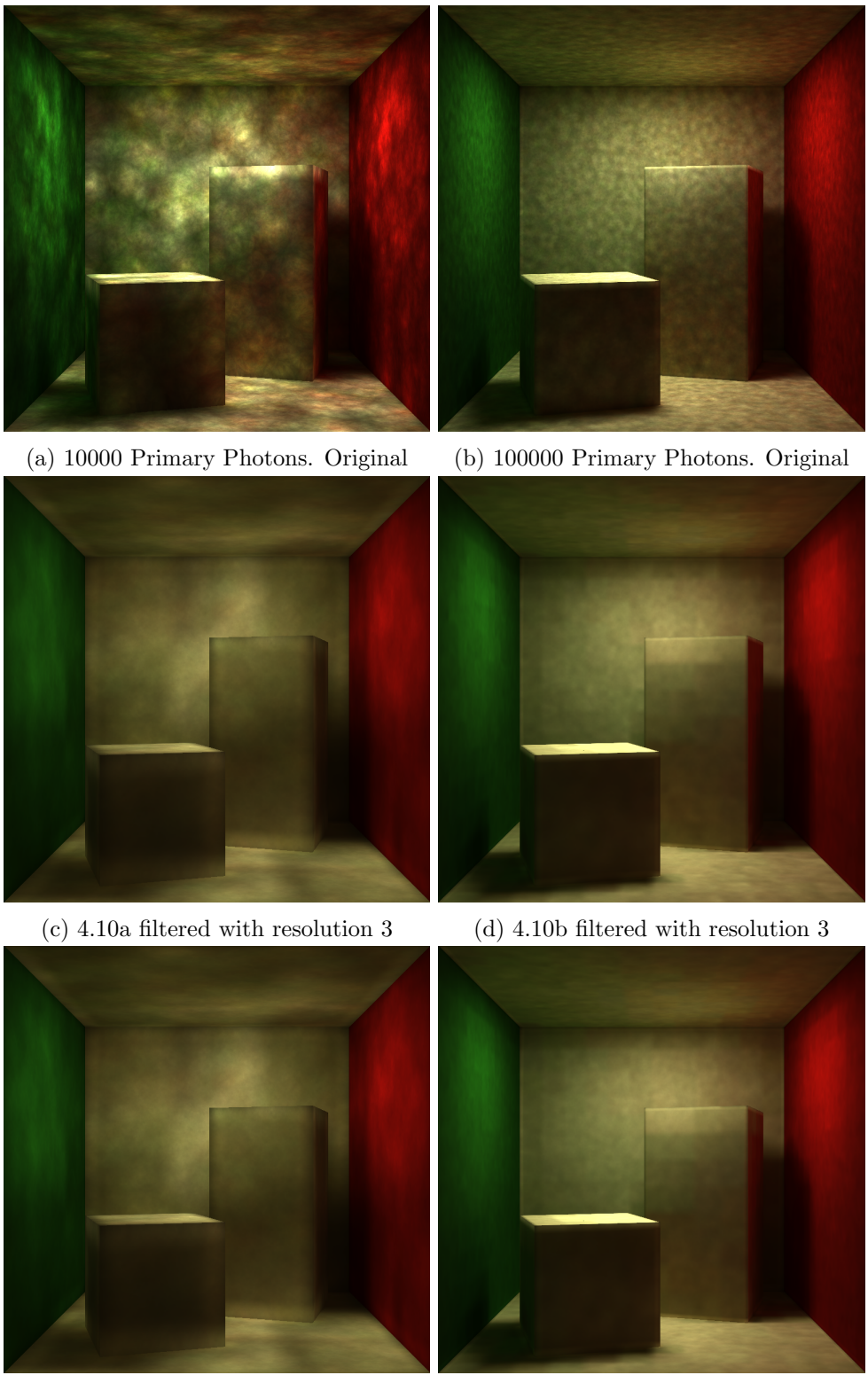


Figure 4.9: A visual comparison of the difference between the ground truth image and the images from figure 4.8

Visual Quality

As stated earlier, making objective studies and comparisons of visual quality of images is beyond this thesis. However, some observations on the resulting images will be shared below.

The images were generated by using descriptors with resolutions 3 and 5, $\alpha = 0.4$ and variable σ^2.



(a) 10000 Primary Photons. Original

(b) 100000 Primary Photons. Original

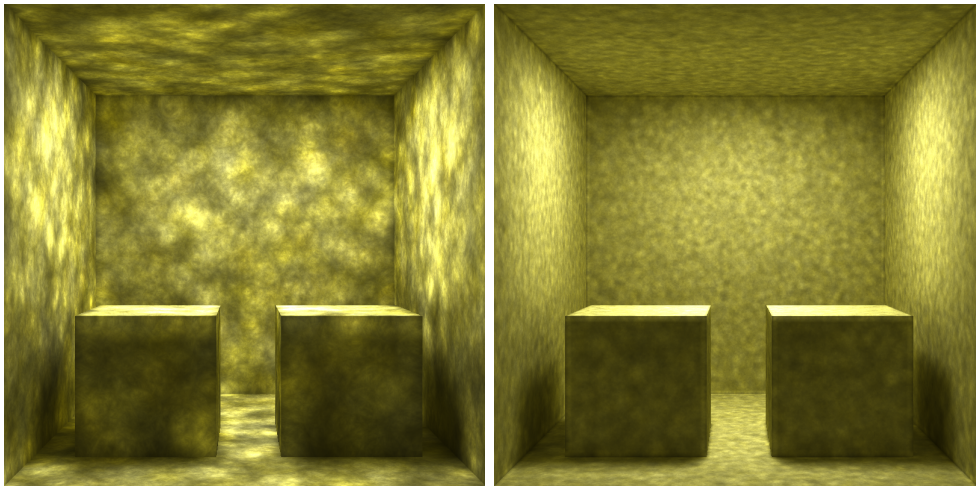
(c) 4.10a filtered with resolution 3

(d) 4.10b filtered with resolution 3

(e) 4.10a filtered with resolution 5

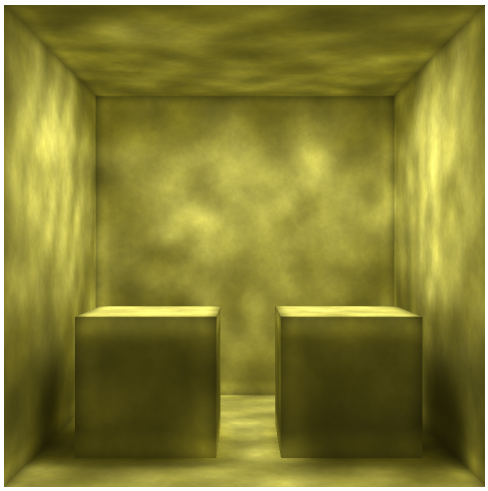
(f) 4.10b filtered with resolution 5

Figure 4.10: Images of the algorithm applied to the Cornell box scene.

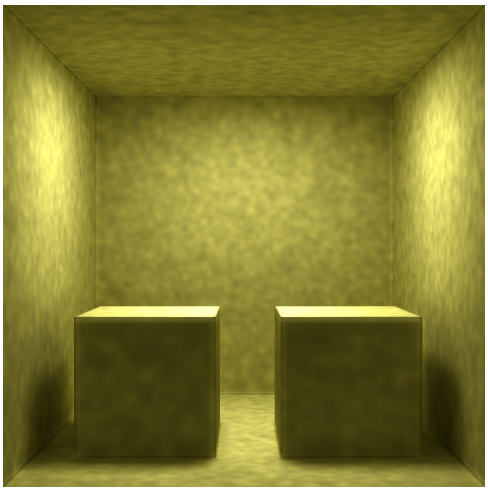


(a) 10000 Primary Photons. Original

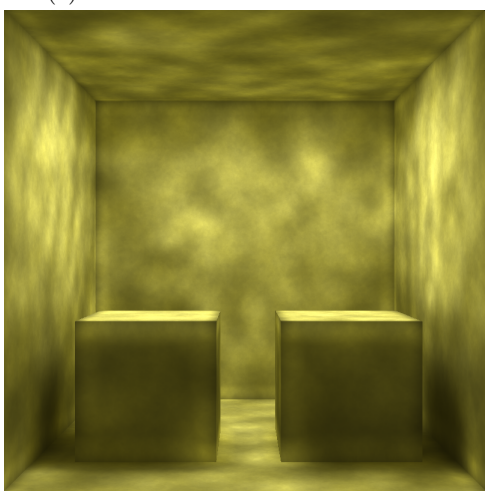
(b) 100000 Primary Photons. Original



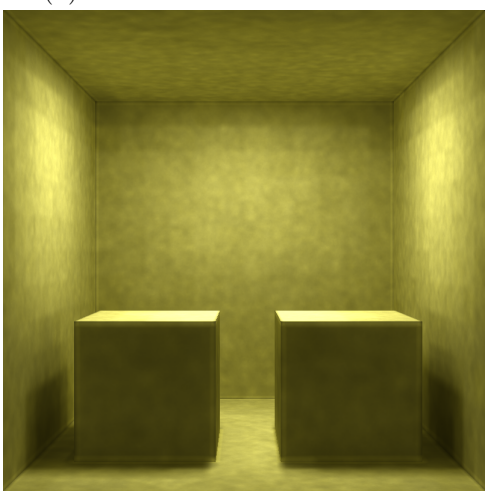
(c) 4.11a filtered with resolution 3



(d) 4.11b filtered with resolution 3



(e) 4.11a filtered with resolution 5



(f) 4.11b filtered with resolution 5

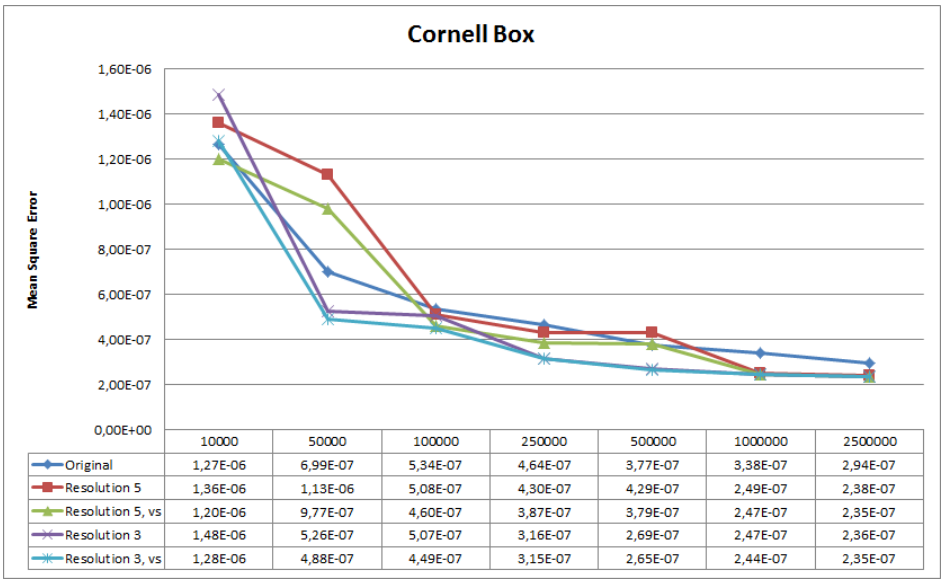
Figure 4.11: Images of the algorithm applied to the aligned Cornell box scene.

Upon visual inspection of the resulting images, one can find that the results seem to follow the theoretical model explained above. When using smaller amounts of primary photons, the

noise removal seems to be much more propounded, than with high photon counts. However, it is worth noting, that a smoother image does not necessarily mean more correct, or closer to the ground truth. Also, as far as our visual inspection permits, the aligned Cornell box scene, which has exaggerated symmetry does not give considerably better results.

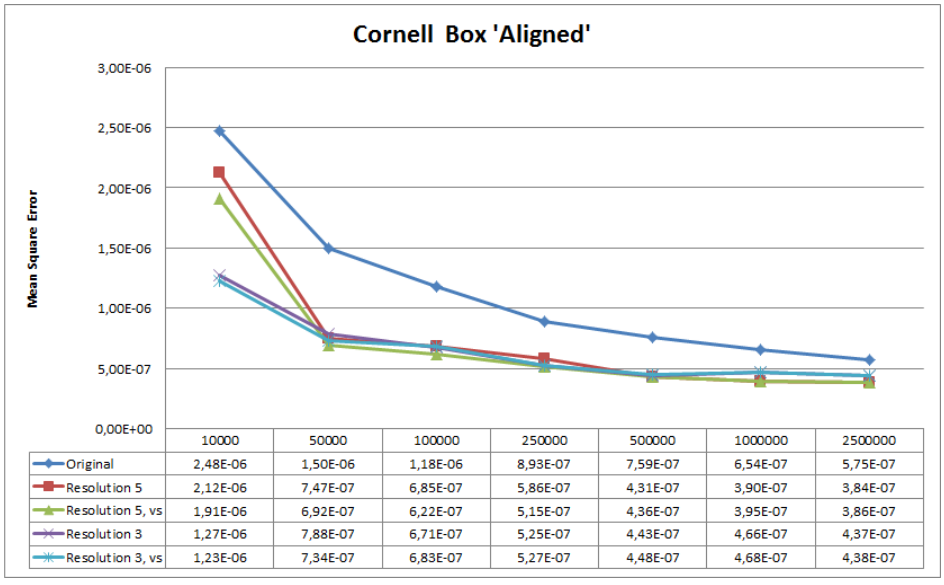
Error

Compared to the unfiltered photon map, we see a decreased error when using smaller resolution, and somewhat increased with higher resolutions. As the primary photon count increases the difference, between the error values gets smaller. The latter correlates with our perdition of the behaviour of the algorithm.



	10000	50000	100000	250000	500000	1000000	2500000
Original	1,27E-06	6,99E-07	5,34E-07	4,64E-07	3,77E-07	3,38E-07	2,94E-07
Resolution 5	1,36E-06	1,13E-06	5,08E-07	4,30E-07	4,29E-07	2,49E-07	2,38E-07
Resolution 5, vs	1,20E-06	9,77E-07	4,60E-07	3,87E-07	3,79E-07	2,47E-07	2,35E-07
Resolution 3	1,48E-06	5,26E-07	5,07E-07	3,16E-07	2,69E-07	2,47E-07	2,36E-07
Resolution 3, vs	1,28E-06	4,88E-07	4,49E-07	3,15E-07	2,65E-07	2,44E-07	2,35E-07

(a) Cornell box scene



	10000	50000	100000	250000	500000	1000000	2500000
Original	2,48E-06	1,50E-06	1,18E-06	8,93E-07	7,59E-07	6,54E-07	5,75E-07
Resolution 5	2,12E-06	7,47E-07	6,85E-07	5,86E-07	4,31E-07	3,90E-07	3,84E-07
Resolution 5, vs	1,91E-06	6,92E-07	6,22E-07	5,15E-07	4,36E-07	3,95E-07	3,86E-07
Resolution 3	1,27E-06	7,88E-07	6,71E-07	5,25E-07	4,43E-07	4,66E-07	4,37E-07
Resolution 3, vs	1,23E-06	7,34E-07	6,83E-07	5,27E-07	4,48E-07	4,68E-07	4,38E-07

(b) Aligned Cornell box scene

Figure 4.12: Mean square error of the algorithm compared to unfiltered photon mapping, vs denotes *variable* σ^2. The test were performed with $\alpha = 0.4$

One interesting aspect is the lack of improvement, in terms of MSE when using the least amount of primary photons - 10000, although the denoising there seems strongest. In some tests filtering produces even higher error than the original photon map. We attribute this to bias incurred by the low amounts of photons. which our algorithm, unfortunately, cannot counter.

Using a higher descriptor resolution seems tor result in degradation of quality. Reason for this might be the fact that higher descriptor resolutions lead to larger segments, as they are subdivided based on the amount of photons required to generate the descriptor. This, in turn leads to more blur in the final render and overall increases the blocking effect.

Using a variable σ^2 results in a small but positive effect on the filtering, especially with smaller photon counts. We suspect that this is due to the larger uniformity in the filtering that the variable filtering parameter provides and thus minimizes blocking when using large segments. Additionally, this allows to use less minimize the number of independent parameters.

Results of the experiments ran on the 'aligned Cornell box' scene are considerably better than the ones done with the ordinary 'Cornell box'. This most probably is due to the abundance of local symmetry and redundancy in the scene and shows that our method successfully takes advantage of such characteristics.

4.2.2 Discussion

Our experiments indicate, that the proposed method performs better that the original photon mapping algorithm. However, the scenes used in our experiments exhibit a lot of redundancy, which possibly enhances the method's performance. Additional tests might be preformed with scenes that have less local symmetry in terms of geometry and colour, but this is beyond the scope of this thesis.

Visually the method removes considerable amounts of noise, especially when used with small to medium photon counts.

Keeping in mind that good timewise performance is not a goal of this thesis rendering times are quite high. The proposed iterative rendering approach increases the time needed to render a scene approximately by a factor of k, where k is the amount of elements used to filter a segment.

Photons	10000	50000	100000	250000	500000	1000000	250000
Render	58.0673	86.0691	113.712	186.914	278.388	472.578	1276.13
Filter	0.188963	1.67871	1.01	1.246	10.9747	10.6867	14.6778

Table 4.1: Average rendering times and filtering times of the tests performed in section 4.2.1. The render times are for 20 iterations.

Artefacts seem to occur when a large amount of segments in filtering. Since nearby segments can be filtered with different sets of elements, a *blocking* effect occurs when using a lot of neighbours for the filtering. The result is illustrated in figure 4.13 Depending on the the value of α artefacts seem to occur when using above 20 elements for filtering.

Figure 4.13: Blocking artefacts occur even at small segment sizes, when the amount of elements used for filtering is high.

Chapter 5

Conclusion

We proposed a method for filtering photon maps in world space as a noise reduction strategy. The method relies on identifying symmetrical geometric segments, which here is done ad-hoc by a regular patch grid, as a necessary condition for the existence of areas in the final image that are symmetric. That is, areas of outgoing radiance that are equal under a certain transformation.

The photon mapping approach proved very useful for the purposes of this method, as it allowed us to work with flux and ignore the complicated recursive dependencies of the illumination in a scene. The approximate solution, generated in the global photon map, allowed to identify areas that besides geometric symmetry share similar incoming flux. The geometric symmetry along with similar incoming light and uniform $BRDF$ served as a prerequisite to areas that exhibit the same outgoing radiance. The photon map is then filtered using a non-local means filter. The photons from similarly lit segments were accumulated and rendered iteratively.

Overall the method showed an improvement in the visual quality of the density estimation of the global photon map. However, when increasing the amount of primary photons and improving the quality of global illumination, the effects of the filter become less propounded. The practical use of the method is also limited, because of the large render times. Given the lack of substantial work in this area, we believe that the method can be a good stepping stone for future research.

5.1 Future Work

- *Genuine geometric symmetry detection:* a significant improvement would be to implement an automatic symmetry detection algorithm. Different classes of symmetrical objects can be used to cross-filter each other. Additionally, an approach that uses incoming illumination to drive symmetry detection would be ideal for this method. A big drawback in the current implementation is that areas of interest, like shadow boundaries or bright spots are rarely aligned with geometric segments. Allowing the method to select symmetrical regions, based on areas of interest in the illumination might dramatically improve visual quality.

- *Caustics:* Rendering caustics is a part of photon mapping that can benefit from our method, since caustics are visualized through directly rendering the radiance estimate. Additionally, caustic photon maps are much higher resolution than the global photon map and exhibit a lot sharp detail. Thus, they might be very well suited for non-local filtering. However, the biggest challenge for filtering caustics would be detecting

similar areas - since they are mostly local phenomena, a predefine subdivision of the scene might not be very useful. Instead, an illumination driven symmetry detection algorithm might prove useful.

- *Final Gather:* As our method creates a sort of a map of the illumination in the scene, this map can be used to potentially cache computed radiance estimate samples. When rendering the global photon map using the approximate solution, radiance estimate samples can be caches at the segments and then looked up from the kd-tree and reused when similar segments are being sampled. Depending on the search speed, compared to the speed of density estimation this could potentially speed-up rendering of the global photon map.

Bibliography

[1] Arthur Appel. Some techniques for shading machine renderings of solids. In *Proceedings of the April 30–May 2, 1968, Spring Joint Computer Conference*, AFIPS '68 (Spring), pages 37–45, New York, NY, USA, 1968. ACM.

[2] James Arvo. Backward ray tracing. In *In ACM SIGGRAPH 86 Course Notes - Developments in Ray Tracing*, pages 259–263, 1986.

[3] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, September 1975.

[4] M. Bokeloh, A. Berner, H. p. Seidel, and A. Schilling. Symmetry detection using feature lines. *Comput. Graph. Forum*, pages 697–706, 2009.

[5] Martin Bokeloh, Michael Wand, and Hans-Peter Seidel. A connection between partial symmetry and inverse procedural modeling. *ACM Trans. Graph.*, 29(4):104:1–104:10, July 2010.

[6] A. Buades, B. Coll, and J. M. Morel. A review of image denoising algorithms, with a new one. *Simul*, 4:490–530, 2005.

[7] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. A non-local algorithm for image denoising. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2 - Volume 02*, CVPR '05, pages 60–65, Washington, DC, USA, 2005. IEEE Computer Society.

[8] Michael F. Cohen, Shenchang Eric Chen, John R. Wallace, and Donald P. Greenberg. A progressive refinement approach to fast radiosity image generation. *SIGGRAPH Comput. Graph.*, 22(4):75–84, June 1988.

[9] Michael F. Cohen and Donald P. Greenberg. The hemi-cube: A radiosity solution for complex environments. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '85, pages 31–40, New York, NY, USA, 1985. ACM.

[10] Robert L. Cook, Thomas Porter, and Loren Carpenter. Distributed ray tracing. *SIGGRAPH Comput. Graph.*, 18(3):137–145, January 1984.

[11] Greg Coombe, Mark J. Harris, and Anselmo Lastra. Radiosity on graphics hardware. In *Proceedings of Graphics Interface 2004*, GI '04, pages 161–168, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2004. Canadian Human-Computer Communications Society.

[12] Mauricio Delbracio, Pablo Musé, Antoni Buades, Julien Chauvier, Nicholas Phelps, and Jean-Michel Morel. Boosting monte carlo rendering by ray histogram fusion. *ACM Trans. Graph.*, 33(1):8:1–8:15, February 2014.

[13] Alexei A. Efros and Thomas K. Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, pages 1033–, Washington, DC, USA, 1999. IEEE Computer Society.

[14] Cindy M. Goral, Kenneth E. Torrance, Donald P. Greenberg, and Bennett Battaile. Modeling the interaction of light between diffuse surfaces. *SIGGRAPH Comput. Graph.*, 18(3):213–222, January 1984.

[15] Toshiya Hachisuka, Shinji Ogaki, and Henrik Wann Jensen. Progressive photon mapping. *ACM Trans. Graph.*, 27(5):130:1–130:8, December 2008.

[16] Wojciech Jarosz, Henrik Wann Jensen, and Craig Donner. Advanced global illumination using photon mapping. In *ACM SIGGRAPH 2008 Classes*, SIGGRAPH '08, pages 2:1–2:112, New York, NY, USA, 2008. ACM.

[17] Henrik Wann Jensen. Global illumination using photon maps. In *Proceedings of the Eurographics Workshop on Rendering Techniques '96*, pages 21–30, London, UK, UK, 1996. Springer-Verlag.

[18] Henrik Wann Jensen and Niels Jrgen Christensen. Photon maps in bidirectional monte carlo ray tracing of complex objects, 1994.

[19] James T. Kajiya. The rendering equation. *SIGGRAPH Comput. Graph.*, 20(4):143–150, August 1986.

[20] Javor Kalojanov, Martin Bokeloh, Michael Wand, Leonidas Guibas, Hans-Peter Seidel, and Philipp Slusallek. Microtiles: Extracting building blocks from correspondences. *Comp. Graph. Forum*, 31(5):1597–1606, August 2012.

[21] Eric P. Lafortune and Yves D. Willems. Bi-directional path tracing. In *PROCEEDINGS OF THIRD INTERNATIONAL CONFERENCE ON COMPUTATIONAL GRAPHICS AND VISUALIZATION TECHNIQUES (COMPUGRAPHICS 93*, pages 145–153, 1993.

[22] Niloy J. Mitra, Leonidas J. Guibas, and Mark Pauly. Partial and approximate symmetry detection for 3d geometry. *ACM Trans. Graph.*, 25(3):560–568, July 2006.

[23] Fabrice Rousselle, Claude Knaus, and Matthias Zwicker. Adaptive rendering with non-local means filtering. *ACM Trans. Graph.*, 31(6):195:1–195:11, November 2012.

[24] Eric Veach and Leonidas J. Guibas. Metropolis light transport. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, pages 65–76, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[25] Li-Yi Wei and Marc Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 479–488, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[26] Shuang Zhao, Miloš Hašan, Ravi Ramamoorthi, and Kavita Bala. Modular flux transfer: Efficient rendering of high-resolution volumes with repeated structures. *ACM Trans. Graph.*, 32(4):131:1–131:12, July 2013.