國立暨南國際大學資訊工程學系

博士論文
Dissertation

群化光子映射
Grouped Photon Mapping

指導教授：陳履恆 博士
研究生：蔡宗志
中華民國 99 年 1 月 31 日

國立暨南國際大學碩（博）士論文考試審定書

資訊工程學系

研究生蔡宗志所提之論文

群化光子映射
Grouped Photon Mapping

經本委員會審查，符合碩（博）士學位論文標準。

學位考試委員會

歐陽明 委員兼召集人

莊永裕 委員

張鈞法 委員

[illegible] 委員

[illegible] 委員

中華民國 98 年 12 月 5 日

博碩士論文電子檔案上網授權書

（提供授權人裝訂於紙本論文書名頁之次頁用）

本授權書所授權之論文為授權人在 暨南國際大學 資訊工程學系 ________組 98 學年度第 一 學期取得 博士 學位之論文。

論文題目： 群化光子映射
指導教授： 陳履恆

授權人： 蔡宗志
簽　名： 蔡宗志　　　　中華民國 98 年 12 月 28 日

誌謝

首先感謝指導教授陳履恆博士多年來的不吝於指導與教誨，在我人生最無助時刻(Orz)，及瀕臨放棄學業下不斷給予鼓勵與信心，使我能夠在苦悶且漫長的博士生涯中有所收穫及成長，最後才能順利完成此博士論文及學位。特別感激父母及家人在背後所提供莫大的精神支持及經濟支援下，讓我能夠心無旁騖且專心致力於學業及研究。另外要特別感謝我的女友小敏，除了多年來無時無刻的陪伴外，也不時在生活上給予幫忙與添加樂趣，讓我能夠適時獲得壓力的紓解及心情的放鬆，而注入一股源源不絕的〝原力〞去應付艱辛挑戰及經歷各種磨難。

在口試期間承蒙歐陽明、莊永裕、陳炳宇與張鈞法教授們在事務繁忙中，仍能抽空前來擔任口試委員，除了感到莫大的榮幸外，並對於論文研究提出寶貴的建議與明確的指正，也在此致上真誠的敬意與由衷的感謝。

最後當然要感謝實驗室的學弟們(fishman、小黃、winner、蛋蛋、沛宇、永富、襄安、pointer 及其他族繁不及備載…)，一起帶著 happy 的心情陪我度過漫長光陰的日子^ ^，也謝謝學弟 shower 給予學位口試時諸多的幫忙。

蔡宗志 敬上　2009/12/25

論文名稱： 群化光子映射

校院系：國立暨南國際大學科技學院資訊工程學系　　頁數：77

畢業時間：99 年 1 月　　學位別：博士

研究生：蔡宗志　　指導教授：陳履恆 教授

論文摘要

光子映射是一種有效率的全域照明視覺化方法，而且能夠廣泛地模擬在各種不同 3D 虛擬場景中的視覺或光學效果，尤其有助於合成光線聚焦效果。然而，由於光子映射採用 nearest-neighbor 密度估計法，所以對於合成結果容易產生包括 proximity、boundary 及 topological 等類型的偏差。另外對於一些較特別的物體，如鑽石等，由於光線在物體內部的行進軌跡很複雜,所以使用傳統光子映射法會造成無法忍受且可見的缺陷。

在本篇論文中，我們提出結合光子映射法與光束概念的群化光子映射架構，藉以改善 nearest-neighbor 密度估計的方式。基於光束傳遞時的射線一致性,我們將所有光子分類成類似於不同光束的光子群,其中每一個光子群中的光子在他們的傳遞路徑上皆具有一致的相交記錄。之後，我們將每一個光子群儲存在一個獨立光子圖中,並且依據每一光子群的光子分佈來維護一個多邊形邊界,以去重建一個類似光束形成的照明區域。而這照明區域形成了一個自然且精確的取樣光子區域，使我們可以在 query point 周圍去作光子搜尋時，藉以過濾在附近的鄰近光子。另外，我們分別運用一種 Level of detail 技術及一種直覺策略，去控制光子群重組順序及過程，以企求達到偏差與雜訊間的平衡。最後，為了取樣效率的考量,我們建構一個由光子群與光子節點所組成的雙層式 kd-tree 結構，而去輔助在 query point 周圍取樣鄰近光子時，能夠以事先排除大部份無關光子群的方式,來減少花費在點與多邊形區域相交測試及搜尋鄰近光子上的時間。

經由實驗結果可以證明我們所採用的方法能夠成功地減少雜訊及偏差，並且移除漏光現象，特別是對於寶石物體所產生的複雜聚焦效果，我們能夠順利地合成出高品

質的影像。

關鍵詞：光子映射,密度估計,視覺化,全域照明

Title of Thesis：Grouped Photon Mapping

Name of Institute: CSIE, CST, NCNU　　Pages: 77

Graduation Time： 01/99　　Degree Conferred： Ph.D.

Student Name： Tsung-Chih Tsai　　Advisor Name： Lieu-Hen Chen

Abstract:

Photon mapping is an efficient global illumination method. It can be generally applied to simulate various visual or optical effects in complex scenes, especially caustics. However, photon mapping adopts the nearest-neighbor density estimation method, the result tends to generate systematic errors including proximity, boundary, and topological bias respectively. In addition, for some special objects such as diamonds, the transmission paths of light rays inside the object are so complicated that it would result in intolerable visual defects using standard photon mapping method.

In this dissertation, we propose a novel architecture, grouped photon mapping, by combing standard photon mapping with the light-beam concept to improve the nearest-neighbor density estimation method. Based on the ray coherence of a light-beam, we cluster all photons into different beam-like groups of photons, where the photons in the same group have a coherent intersection-history in their transmission paths. We store each photon group in an isolated photon map which is also used to maintain a polygonal boundary to rebuild a beam-like illuminated area by the distribution of the photons. The illumination area forms a natural and accurate sampling area to filter the neighbor photons around the query point in the nearest-neighbor query stage. In addition, we apply a Level of Detail technique and an intuitive strategy for managing the process of reorganizing groups to achieve a trade-off between bias and noise. Finally, in terms of efficiency, we construct a double-layer kd-tree consisting of group and photon nodes respectively. This data structure is useful to first exclude most of unrelated photon groups and reduce the time on the point-boundary intersection testing and searching for the nearest-neighbor photons.

The experiment results prove that our method can successfully reduce noise and bias, and eliminate light leakage. Especially, for a gemstone object with complicated caustic effects, we can smoothly synthesize a high-quality image.

Keywords：Photon Mapping, Density Estimation, Visualization, Global Illumination

Table of Contents

誌謝 I

論文摘要 II

Abstract IV

Table of Contents VI

List of Figures IX

List of Tables XI

1.Introduction 1

1.1. Motivation 1

1.2. Research Goal 5

1.3. Dissertation Organization 6

2.Literature Review 7

2.1. Bandwidth Decision 7

2.2. Area Estimate 7

2.3. Filter Function 8

2.4. Nearest-Neighbor Query 9

3.Fundamental Background 12

3.1. Photon Mapping 12

3.1.1. Overview 12

3.1.2. Russian Roulette 13

3.1.3. Photon Maps 14

3.1.4. Photon Structure 16

3.1.5. The Balanced Kd-Tree 17

3.1.6. Radiance Estimate 18

3.1.7. Rendering 20

3.1.8. Direct Illumination 21
3.1.9. Specular Reflection 22
3.1.10. Caustic Illumination 23
3.1.11. Soft Indirect Illumination 24
3.2. Level of Detail 25
3.2.1. Introduction 25
3.2.2. Polygonal Simplification using Quadric Error Metrics 27
3.2.3. Candidate Vertex Pairs 29
3.2.4. Quadric Error Metrics 29
3.3. Diamond Optics 32
3.3.1. Total Reflection 34
3.3.2. Dispersion and Fire 35
3.3.3. Geometry Cutting 36
4. Grouped Photon Mapping 37
4.1. Basic Concept 38
4.2. Overview 38
4.2.1. Photon Pass 39
4.2.2. Rendering Pass 42
4.3. Merging Groups using Quadric Error Metrics 43
4.4. Polygonal Boundary 46
4.5. Radiance Estimate 47
4.5.1. Advanced Nearest-Neighbor Density Estimation 47
4.5.2. Double-Layer Kd-tree 50
4.6. Visualizing Global Illumination using Grouped Photon Maps 52
5. Results and Discussion 54
5.1. System and Environment Configuration 54
5.2. Level of Detail Test 54
5.3. Light Leakage Test 60
5.4. Diamond Caustics Test 62

5.5. Run-Time Statistics 69
6. Conclusion and Future Work 71
6.1. Conclusion 71
6.2. Future Work 72
Bibliography 74

List of Figures

Figure 1.1. *Proximity bias* leads to a worse result for a bigger searching area. 2

Figure 1.2. The realistic photograph of a crystal-like gemstone with complex caustics...... 2

Figure 1.3. A synthesized image is rendered by standard *photon mapping* method. 3

Figure 1.4. *Boundary bias*. 3

Figure 1.5. *Topological bias*... 4

Figure 1.6. Light leakage. 4

Figure 3.1. Caustic photons. 15

Figure 3.2. Direct photons. 15

Figure 3.3. An indirect photon... 16

Figure 3.4. Construct a two dimensional kd-tree.. 17

Figure 3.5. The reflected radiance L_r.. 20

Figure 3.6. Specular reflection.. 23

Figure 3.7. The *final gathering* technique. 24

Figure 3.8. The bunnies are represented by different versions of *level of detail*............... 26

Figure 3.9. The process for contracting a vertex-pair.. 28

Figure 3.10. An unconnected vertex-pair (v_i, v_j). 28

Figure 3.11. The cutting rate of a round brilliant cut... 33

Figure 3.12. Total reflection. 33

Figure 3.13. Reflective intensity v.s. Incident angle. 34

Figure 3.14. A dispersed light-beam splits apart and spreads out in different directions... 35

Figure 3.15. Emerald-cut and marquise-cut. 36

Figure 3.16. Oval-cut and pear-cut.. 36

Figure 4.1. A reflected or refracted light-beam strikes the table. 37

Figure 4.2. The system architecture for *grouped photon mapping*................................... 39

Figure 4.3. A photon is generated by performing *uniform random sampling*.................... 40

Figure 4.4. An intersection history. 41

Figure 4.5. Three discrete photons (represented by green color rays) are clustered. 41

Figure 4.6. Using QEM to control the process of merging groups.................................... 45

Figure 4.7. The processes of building a polygonal boundary. 46
Figure 4.8. Performing the point-boundary intersection testing at query point x............... 48
Figure 4.9. The point-boundary intersection testing. 49
Figure 4.10. Locate 10 closest photons (red color) to the query point. 50
Figure 4.11. The double-layer kd-tree. 51
Figure 5.1. The caustic effect is rendered before performing any surface simplification. . 55
Figure 5.2. The caustic effect is rendered after reducing 15% of vertices. 56
Figure 5.3. The caustic effect is rendered after reducing 30% of vertices. 57
Figure 5.4. The caustic effect is rendered using standard *photon mapping*. 58
Figure 5.5. Valid-photon (%) vs. Reduced-Vertex (%) for the glass sphere scene.. 59
Figure 5.6. Valid-photon (%) vs. Reduced-Vertex (%) for the glass bunny scene.. 60
Figure 5.7. Light leaks from left room into right room while using *photon mapping*........ 61
Figure 5.8. There is no light leakage while using *grouped photon mapping*. 62
Figure 5.9. A round brilliant cut of diamond with complex caustics (PM). 64
Figure 5.10. A round brilliant cut of diamond with complex caustics (GPM). 65
Figure 5.11. A round brilliant cut of diamond with complex caustics(*Path tracing*). 66
Figure 5.12. Emerald-cut diamond is rendered using *grouped photon mapping*. 67
Figure 5.13. Pear-cut diamond is rendered using *grouped photon mapping*. 68
Figure 5.14. Multiple diamonds are rendered using *grouped photon mapping*.................. 69
Figure 6.1. A visible dark edge appears around the corner. 73

List of Tables

Table 3.1. Three types of photon maps 14
Table 5.1. Run-time statistics. 70

1. Introduction

1.1. Motivation

Synthesizing photo-realistic CG images is a quite interesting and challenge task. Since 1970, it has been being one of most important research issues in the field of *computer graphics*. There are a great number of global illumination techniques developed to handle and simulate various optical effects, such as ray tracing based methods [1][2][3][4], radiosity based methods [5][6], and hybrid methods [7][8][9]. As a successor, the *photon mapping* (PM) method by Jensen [10] is a popular and important global illumination algorithm. It successfully integrates the predecessors to deposit illumination information in an independent map from the object geometry. This enables us to simulate various optical phenomena well in the scenes with highly complex models.

However, as a result of *photon mapping* adopting the nearest-neighbor density estimation method, it suffers from the problem of systematic error. As Schregle mentioned in his research [11], the sources of error are mainly divided into three types: *proximity, boundary,* and *topological bias*.

Proximity bias is inherent to the *photon mapping* method, which uses a non-zero-sized searching area to locate the *N* closest photons around a query point. (see Figure 1.1) In this case, the query point is a point on a surface to evaluate the illumination contribution. This bias often leads to blurry and visible defects near the illumination features, such as the edge of caustics.

To compensate for the *proximity bias*, several filter functions have been applied to the nearest-neighbor density estimation method, such as the Cone [12], 2D Epanechnikov [13], and Silverman kernel functions [14]. Depending on the distance relationship between the photon and the query point, the filter gives more weight to the closer photons to the query point. Although this can reduce the bias and get a smoother result in general scenes, the result is still insufficient in certain special cases. A good example is shown in Figure 1.2

which is a realistic scene with a crystal gemstone inside. There are a great number of caustics with sharp and hard edge accompanying these gemstones. In comparison with Figure 1.3, which is rendered by standard *photon mapping* method, the differences on the shape and edge of caustics are very clear and noticeable.

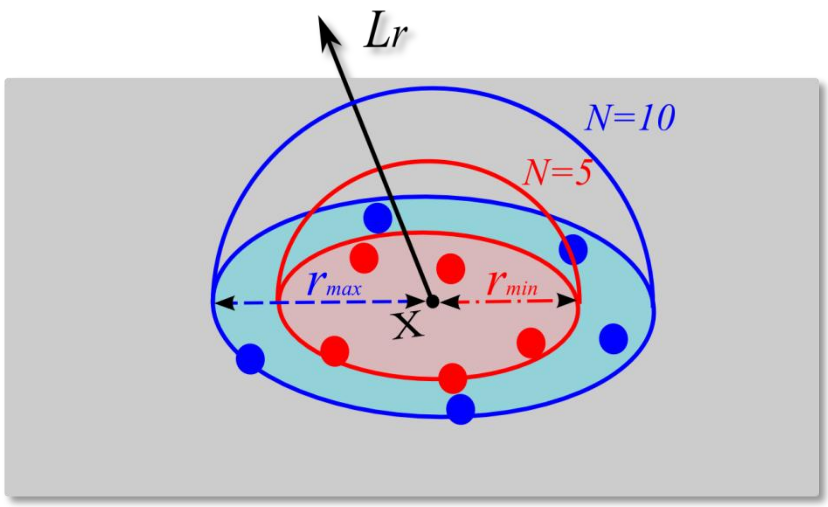

Figure 1.1. *Proximity bias* leads to a worse result for a bigger searching area.

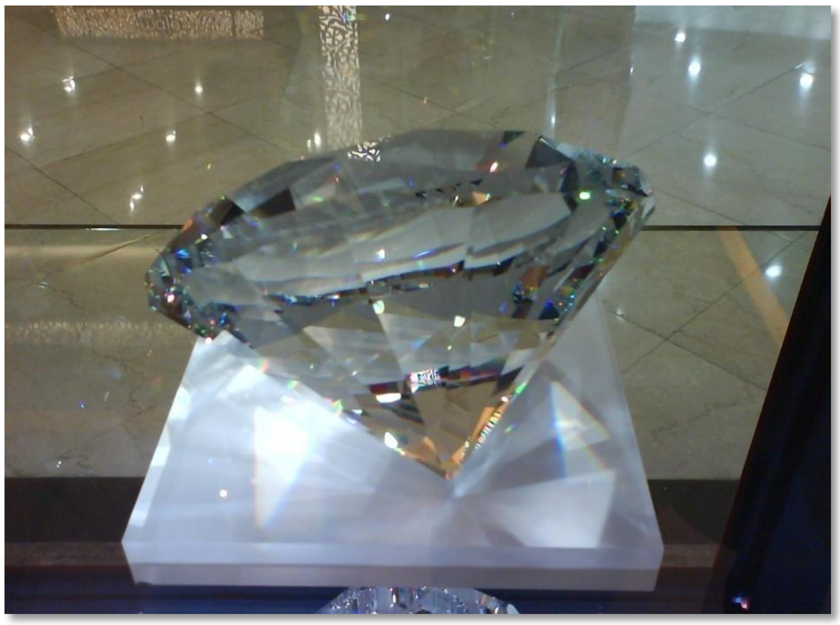

Figure 1.2. The realistic photograph of a crystal-like gemstone with complex caustics

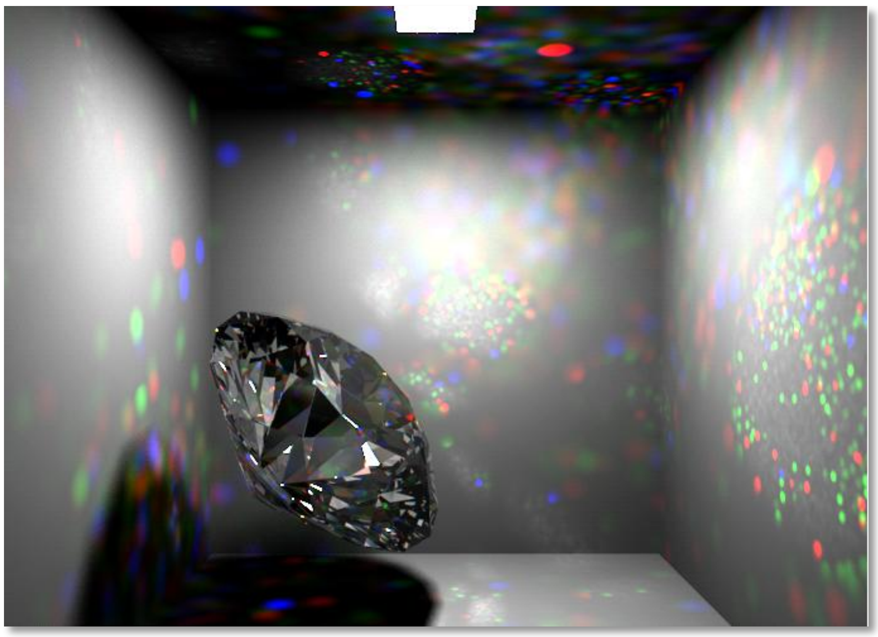

Figure 1.3. A synthesized image is rendered by standard *photon mapping* method.

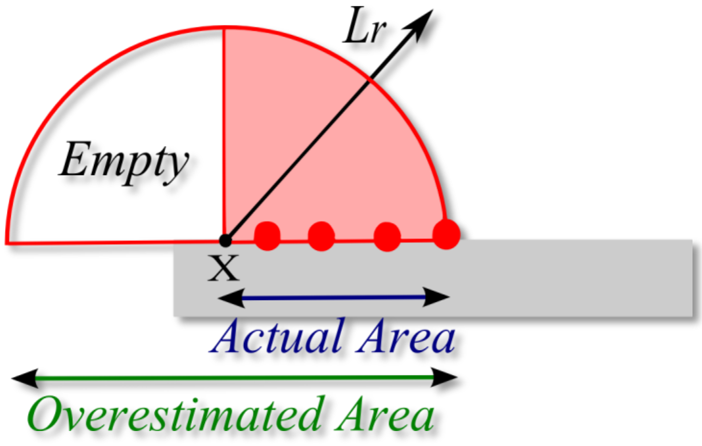

Figure 1.4. *Boundary bias* leads to an overestimated area near the border.

Both *boundary* and *topological bias* originate from evaluating inaccurate area estimates by using the nearest-neighbor density estimation method. The former overestimates the area as shown in Figure 1.4. This problem results in low energy-density, and therefore often generates inaccurate and dark images. *Boundary bias* happens to the border of an object which the query point approaches. Conversely, the latter underestimates the area, as shown in Figure 1.5, and has a high energy-density such that a brighter result is generated. *Topological bias* appears on a non-planar surface in which the query point is situated.

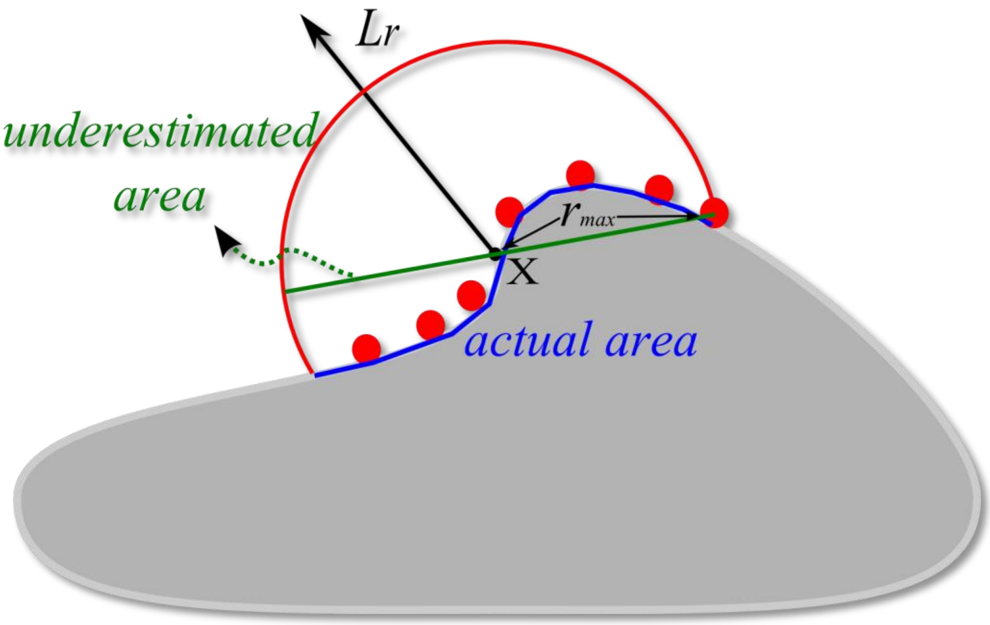

Figure 1.5. *Topological bias* leads to an underestimated area on the non-planar surface.

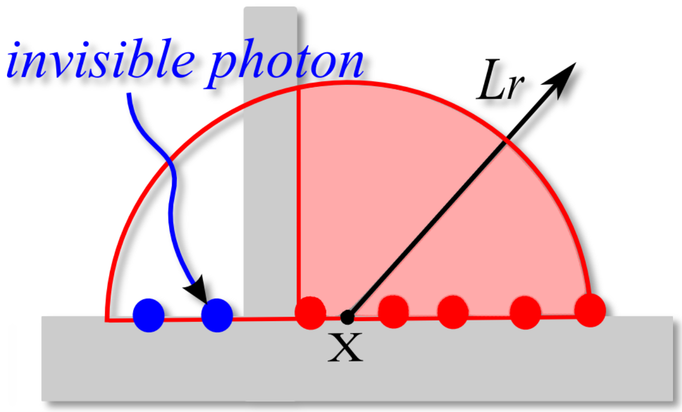

Figure 1.6. Light leakage is generated from the invisible photons behind an obstacle located by the query point.

Though *photon mapping* has the advantage of decoupling the photon storage from the object geometry, this would also lead to lose the local visibility of photons to the eyes. This problem often causes the query point to sample invisible photons and results in light leakage. Figure 1.6 illustrates a noticeable situation, where light leaks from an wall's backside due to mis-locate invisible photons.

Based on the problems from the nearest-neighbor density estimation method, in this dissertation, we propose a new solution to reduce bias and eliminate light leakage.

1.2. Research Goal

The purpose of this dissertation is to solve bias and light leakage problems from standard *photon mapping* using a simple expandable searching area like a sphere or disk to mis-locate false or inappropriate photons. This method is mainly applied to the scene which consists of polygonal models. In order to enhance the nearest-neighbor density estimation method, we present an advanced method by exploiting a more precise and adaptive sampling area to filter neighbor photons around the query point. This method is effective to reduce bias and eliminate light leakage. In addition, it can be also applied to synthesize high-complexity caustics generated from a gemstone which cannot be correctly rendered by standard *photon mapping*.

In this dissertation, we present a new approach, *grouped photon mapping* (GPM), which mainly extends the standard *photon mapping* method by Jensen [10] and incorporates the light-beam concept to build a nature and adaptive searchable area for the query point. Based on the coherence of intersection history, we cluster discrete photons into different groups of photons. This enables us to rebuild a beam-like illumination area from each group by constructing a polygonal boundary according to the distribution of photons. Finally, in the rendering pass, the illumination area will be as a nature filter to restrict the searchable area to sample photons around a query point. By detecting which the boundaries of groups are struck by the query point, we can accurately locate the *N* nearest-neighbor photons from the struck groups and exclude false photons that are invisible to the eyes or outside the illumination feature.

1.3. Dissertation Organization

The remainder of this dissertation is organized as follows: Chapter 2 reviews the previous work on the *photon mapping* method. Chapter 3 describes the fundamental background including *photon mapping*, *level of detail*, and diamond optics. In Chapter 4 we details our *grouped photon mapping* algorithm from introducing basic concept and overview to explain how to merge noise groups using quadric error metrics, create polygonal boundary from each photon group, and perform radiance estimate by advanced nearest-neighbor density estimation with double-layer kd-trees. Besides, the experimental results and discussion are shown in chapter 5. Finally, we conclude the thesis and some problems remains for future work in chapter 6.

2. Literature Review

Many literatures have been proposed to improve standard *photon mapping* method. They were devoted to compensating for the defects of bias. Based on the types of research approaches, we divide them into four sections: Bandwidth Decision, Area Estimate, Filter function, and Nearest-neighbor Query.

2.1. Bandwidth Decision

In an early research by Jensen [15], he adopted a *differential checking* method to avoid blurring the edge of a sharp illumination area such as a caustic. This method dynamically determines the available number of nearest-neighbor photons around a query point by observing the variation of radiance estimates near the edge based on different number of photons. An available estimate will be adopted when the estimate is constantly either increasing or decreasing as adding more photons progressively. However, this approach tends to generate noise near the edge and is hard to control the process of radiance estimate.

Schregle [11] combined the aspects of Jensen's and Walter's approaches to present a *bias compensating operator* for the nearest-neighbor density estimation. The operator evaluates the deviations between noise and bias to estimate the irradiance and determine an optimal bandwidth (the number of photons). Although the *bias compensating operator* can achieve a better result and trade-off between noise and bias, it is quite time-consuming and relies on users to manually specify some adequate bandwidth parameters. In addition, the operator is unsuitable to handle the topological bias because of a lack of the geometry surfaces photons reside on.

2.2. Area Estimate

In the nearest-neighbor density estimation method, some papers were devoted to improve the area estimate which computes the area containing all of *N* nearest-neighbor photons around a query point. The conventional *photon mapping* method adopted an expandable sphere to locate the *N* nearest-neighbor photons. Therefore, the area estimate is to compute the circular projected area by projecting the sphere onto the surface where the photons are stored. However, this approach not only tends to locate false photons, but also overestimate an area for the query point which is close to corners or the edge of objects. Consequently, this overestimation would lead to a boundary bias result. Although later Jensen [16] suggested constructing a minimum-area convex hull which encloses the photons to replace the circular projected area, this way does not have efficiency and deal with the *topological bias* problem.

Hey and Purgathofer [17] exploited an axis-aligned cube, which centers around the query point, to locate the *N* nearest-neighbor photons instead of a conventional sphere or disc. They estimated the target areas by calculating the internal intersection area between the cube and the surface polygons hit by the located photons. This method can deal with the problem of overestimating the area and give a smoother radiance estimate near the edge or corners of objects. However, it needs taking a great deal of time to perform the cube-polygon intersection testing on the radiance estimate.

Tobler and Maierhofer [18] constructed eight-sized bounding boxes (octoboxes) which contains all of *N* nearest-neighbor photons to decrease the problem of overestimating the area. In addition, to eliminate light leakage, they shot 4 or 8 geometry feelers to create a bounding box which centers around the query point so as to exclude the false photons outside the box. However, this method requires a great amount of time for intersection testing between the rays and the neighbor geometry objects. Moreover, for more complicated scenes, the method to shoot limited feelers is insufficient to effectively eliminate light leakage.

2.3. Filter Function

Jensen [12] proposed a *cone filter* to be used on the radiance estimate. This is helpful to reduce the blurry edge of a sharp illumination area such as caustics. The filter assigns a weight to each photon which is located or queried by a query point. The weight ω_p, as shown in Equation 2.1, is based on the distance (d) relationship between a photon and the query point to give the closer photons higher weights.

$$\omega_p = 1 - \frac{d}{kr}$$

(2.1)

Where k >=1 is a filter constant and r is the maximum searching distance. ω_p is then substituted into the filter function $K(p)$ as follows:

$$K(p) = \frac{\omega_p}{\pi r^2 \left(1 - \frac{2}{3k}\right)}$$

(2.2)

Walter [13] and Shirley *et al.*[14] also utilized *Epanechnikov* and *Silverman kernel function* as showed in Equation 2.3 and Equation 2.4 respectively. Especially, the latter could obtain a better and smoother result than both the former and the *cone filter*. Therefore, in this dissertation we also adopt *Silverman kernel function* on the radiance estimate.

$$K(p) = \frac{2}{\pi r^2}\left(1 - (\frac{d}{r})^2\right)$$

(2.3)

$$K(p) = \frac{3}{\pi r^2}\left(1 - (\frac{d}{r})^2\right)^2$$

(2.4)

2.4. Nearest-Neighbor Query

In a paper by Lastra et al. [19], the algorithm showed that the photon map stores more photon information, including the first and second intersections of all ray segments, the origins, and the vectors, than the conventional photon map. On the nearest-neighbor photon query, a disc-shaped searching area, instead of a sphere area, is used to locate the *N* nearest-neighbor photons around the query point. While any neighbor photon is queried and sampled, it must satisfy the following conditions. Either the first or second intersection point must reside in the disc area. This way can effectively reduce *topological bias*. In addition, when a disc area is close enough to the corners or walls of a scene, the actual area of a reachable region by ray segments is re-evaluated. This estimates a more accurate and smaller area to avoid generating *boundary bias* resulting from overestimate the area.

Havran et al. [20] further extended the method developed by Lastra et al., and also constructed an independent *Ray Map* to store all of the photon paths. Four different kinds of expandable shapes are combined to locate the nearest-neighbor photons around the query point, including disc, hemisphere, sphere, axis-aligned bounding box. Also, three types of nearest-neighbor queries based on the distance metrics are supported to determine which photons are the closest to the query point.

Both methods of Lastra et al. and Havran et al. need tremendous memory space to record the path records of photons and take time to perform the ray-shape intersection testing. In addition, in order to prevent *boundary* and *topological bias*, they used a larger size of sampling area to include more photons near the edge or on curved surfaces of objects. However, relatively it could raise *proximity bias* and generate light leakage because of locating false photons which are invisible to the query point.

Herzog et al. [21] proposed a new architecture using the photon splatting technique. On each photon, they adopted a *conical frustum area*, which is controlled by the kernel width, to sample nearby eye-samples which had been first deposited in an earlier stage. Then, through the normalized kernel function, the energy contributing to these eye-samples is calculated and accumulated in the temporary *radiance map*. Although this method efficiently reduces most of bias and light leakage, we need to tune more parameters heuristically for the bandwidth selection and the filter, and consider any exceptional conditions. In addition, this method requires enormous memory resources to record the *radiance map*.

Recently, both *progressive photon mapping* [22] and *stochastic progress photon mapping* [23] extend standard *photon mapping* to propose a multi-pass algorithm. Their first passes are *ray tracing* and *distributed ray tracing* starting from the camera respectively. The following passes are any number of photon tracing passes where each pass showers a fixed number of photons from light sources into the scene and re-locates the *N* nearest-neighbor photons around each eye-sample. The radiance estimate will finally converge to a more correct result by iteratively performing photon passes to increase the number of nearest photons and reduce the searching radius concurrently. A major advantage is unnecessary to store all of photons in the photon map. This enables us to use a limited amount of memory to compute a global illumination solution with any desired accuracy. However, relatively, it needs a large amount of time to estimate accurate radiance. Moreover, unless the searching radius is small enough, it could still result in light leakage or blurry effect near the edge of an illumination feature because of not entirely excluding the false photons outside the illumination area.

3. Fundamental Background

In this chapter, we introduce some related concepts and techniques which this dissertation exploited and extended. This chapter is divided into three sections: *Photon Mapping*, *Level of Detail*, and Diamond Optics. The first section describes the standard photon map method by Jensen [16] which is a global illumination technique. The second section gives an overview of *level of detail* and *quadric error metrics* by Garland and Heckbert [24]. Finally, in the third section, we explain the basic geometrical and optical properties of diamond which will be simulated and rendered in the experimental scenes.

3.1. Photon Mapping

3.1.1. Overview

Photon mapping, which has been presented by Jensen [15] for over ten years, is a well-known two-pass Monte-Carlo *ray tracing* method. The purpose is to develop an independent data structure, the photon map, to deposit the information of indirect illumination which is distributed on non-specular surfaces. The map decouples the representation of the illumination from the object geometry, and makes the photon mapping method possible to handle any geometric and complicated models. In addition, due to the usage of the kd-tree [25] [26] [27] and the nearest-neighbor density estimation method [28], *photon mapping* can be more efficient to synthesize high-quality images than general Monte-Carlo *ray tracing* methods.

The first pass, photon tracing, which is also known as light *ray tracing* or backward *path tracing* [29], is to record the indirect illumination distributed in the scene from light sources. Firstly, a great number of photon particles with carrying the same power (flux) are generated from the uniform random sampling on light sources. Then the photons are emitted and traced through the scene like conventional *ray tracing* method. The difference is that photons are to propagate flux whereas rays are to gather radiance. When a photon hits an object, it can only either to be reflected, refracted or absorbed because of adopting the *Russian roulette* technique [30] to probabilistically decide the interaction type. In addition, the photons which hit non-specular surfaces will be stored in global and caustic photon maps. The photon map organizes and maintains proximity within the stored photons by adopting an efficient kd-tree data structure.

In the second or rendering pass, Jensen directly utilized the photon maps to visualize the indirect and caustic illumination on diffuse surfaces. A nearest-neighbor density estimation method was adopted to estimate radiance by locating the *N* nearest-neighbor photons around the query point. Moreover, in order to reduce the blurry edge of an illumination region, the *cone filter* is used to filter photons so that the nearer photon to the query point has a higher weight on the power than the further one.

3.1.2. Russian Roulette

Russian roulette was first used in *computer graphics* by Arvo and Kirt [30]. It is a stochastic technique which uses a probabilistic sampling way to determine whether a traced photon should be reflected, refracted, or absorbed.

In implementation, for an example, we assume that there is a reflective surface that is hit by a photon. This surface has a diffuse reflection coefficient d, and specular reflection coefficient s, moreover, the sum of d and s is below or equal to one. In addition, a uniformly distributed random variable ξ, whose value is between zero and one, is used to make below decision:

a. $\xi \in [0, d] \Rightarrow \text{diffuse reflection}$

b. $\xi \in]d, d+s] \Rightarrow \text{specular reflection}$

c. $\xi \in]d+s, 1] \Rightarrow \text{absorption}$

Depending on the random value ξ, when ξ corresponds to case a, this means that the photon will be diffusely reflected off the surface. Relatively, if $d < \xi \leq s$, the photon performs specular reflection. Otherwise, if $(d+s) < \xi \leq 1$, the photon is absorbed.

This method ensures that a photon can be quickly terminated in a finite number of bounces. Also, only a fewer photons are necessary to be stored in the photon map. This is because each time a photon as *path tracing* [29] picks a path to bounce rather than spawns several sub-photons to do at the intersection point. Consequently, the radiance estimate is still a correct and unbiased result.

3.1.3. Photon Maps

The photon map is a data structure created to store the photons that hit diffuse surfaces rather than specular surfaces. The reason for not storing photons on specualr surfaces is that the probability of incoming photons matching with the viewer's specular direction is almost zero. In addition, a photon might be stored several times because of bouncing many times on different non-specular surfaces.

For considering the trade-off between quality and efficiency, there are three types of photon maps used to store individual photons based on the difference on the photon path:

Table 3.1. Three types of photon maps

Type	Path Notation	Description
Caustic photon map	LS^+D	Store the photons which have performed at least one specular reflection before hitting a diffuse surface as shown in Figure 3.1.
Global photon map	$L\{S\|D\|V\}^*D$	Store all of the photons from direct, shadow, indirect and caustic illumination before hitting a diffuse surface, as showed in Figure 3.2 and Figure 3.3.
Volume photon map	$L\{S\|D\|V\}^+V$	Store the photons from the indirect illumination of a participating medium before a volume scattering.

The above path notation is expressed by the grammar of Heckbert [39]. *L* means an emission from light sources. *S* is a specular reflection or transmission, while *D* represents a non-specular reflection or transmission. *V* is a volume scattering. The symbol | equals to the logical operator "or". + means that either of *S*, *D* or *V* repeats at least one time. * means that either of *S*, *D* or *V* does not appear or repeats at least one time.

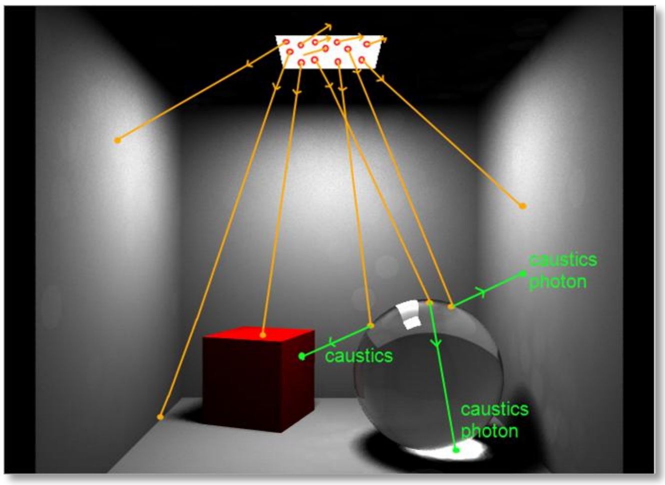

Figure 3.1. Caustic photons which first bounce on any specular surface at least once before being stored on a diffuse surface.

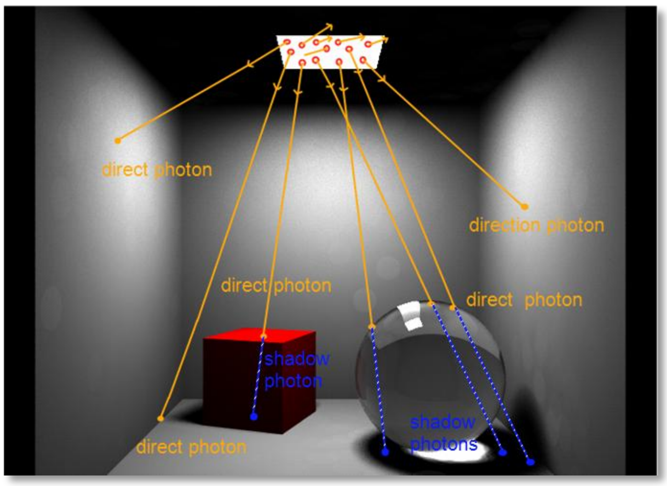

Figure 3.2. Direct photons are stored on the surface which photons first hit, while shadow photons are stored on the next object surface which faces light sources. In which case, the next object is the second object hit by

a photon along the direct photon's direction.

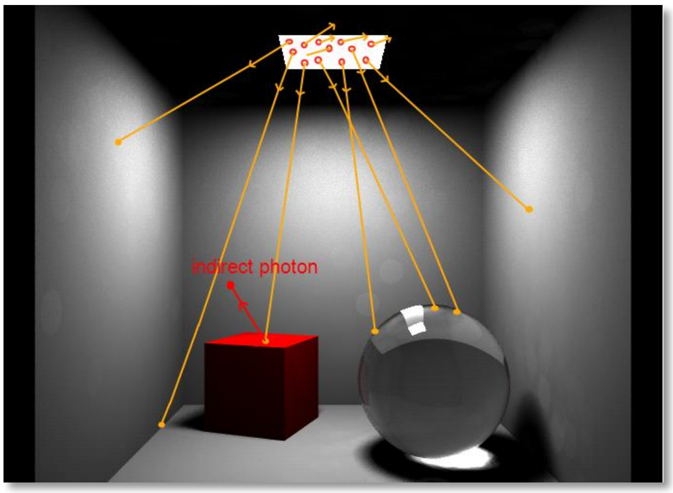

Figure 3.3. An indirect photon which first hits any non-specular surface at least once before being stored on a diffuse surface.

3.1.4. Photon Structure

The required data for storing a photon includes the position, incoming power, incident direction, and flag as shown below [16]:

```
Struct photon {
    float x, y, z;        // intersection position
    char p[4];            // incoming power packed as 4 chars
    char phi, theta;  // compressed incident direction
    short flag;           // flag used in kd-tree
}
```

Where the power data is represented by Ward's RGB-format [31] with 4 bytes which uses an extra 8-bit exponent. The incident direction is denoted by a pair angles, ϕ and θ, of the spherical coordinates and compressed to 65,536 possible directions by the following computation:

$$\phi = 255 * \frac{\text{atan} *(dy,dx)+PI}{2*PI} \tag{3.1}$$

$$\theta = 255 * \frac{\mathrm{acos}(\mathrm{dx})}{\mathrm{PI}} \tag{3.2}$$

3.1.5. The Balanced Kd-Tree

The kd-tree is a data structure for organizing and managing the points in a *k*-dimensional space. The data structure is to construct a binary tree consisting of the points by recursively partitioning a space into two sub-spaces. In other words, it is a special case of *binary space partition tree* (*BSP-tree*) [32]. The purpose of the kd-tree is to accelerate the process of accessing spatial datum. Each tree node represents a point or data item. Moreover, every non-leaf node also plays the role of the position of a splitting plane or hyperplane which divides the space into two subspaces. Depending on different ways to construct the kd-tree, its structure may become balanced or very skewed. As a result, the average time-complexity for searching a point in the kd-tree is $\mathrm{O}(\log_2 \mathrm{N})$ and $\mathrm{O}(\mathrm{N})$ respectively. So, for efficiency, it is quite natural to construct a balanced kd-tree for querying the *N*-nearest neighbor photons.

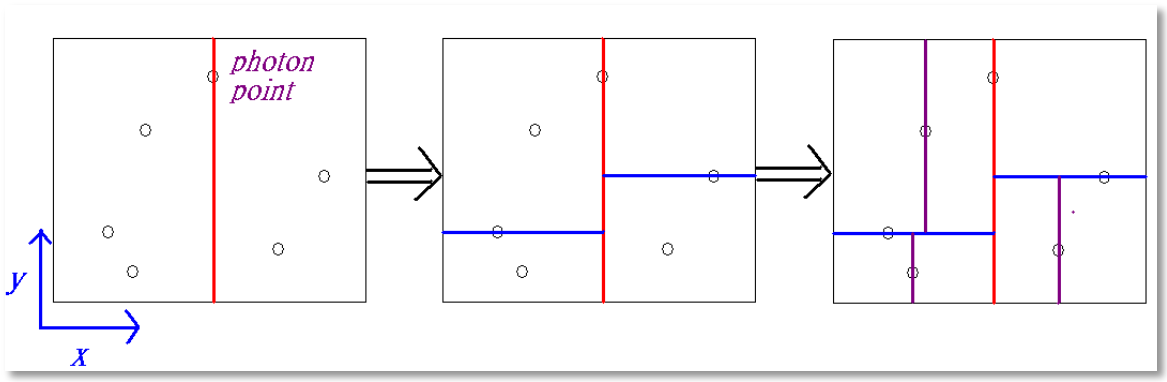

Figure 3.4. Construct a two dimensional kd-tree: (a) Sort the points, choose the median point to be as the root, and partition the x-axis. (b) Recursively sort left and right subspaces, choose the new left and right sub-nodes, and partition the subspaces. (c) Process the remainder points until all points have been put in the tree.

Figure 3.4 illustrates the stages of constructing a balanced kd-tree. Firstly, we have given *N* photons in the space and then sort them in the x-axis value order. Later, we choose the median point from the sorted list to be as the root node of the tree. Also, we split the x-axis into left and right subspaces along the position of the median point. After that, we recursively continue in each half space to sort the remainder points, allocate the new sub-node, and split the subspace, until all points have been processed and put in the tree.

3.1.6. Radiance Estimate

Before visualizing global illumination using the photon maps, we need to first understand how to compute the reflected radiance (L_o) at a given intersection point x on the surface. This can be done by evaluating the following light transport equation (derive from *the rendering equation*) [29]:

$$\begin{aligned} L_o(\mathrm{x}, \omega_o) &= L_e(\mathrm{x}, \omega_o) + L_r(\mathrm{x}, \omega_o) \\ &= L_e(\mathrm{x}, \omega_o) + \int_\Omega f_r(\mathrm{x}, \omega_i, \omega_o)\, L_i(\mathrm{x}, \omega_i) |\cos\theta_i| d\omega_i, \end{aligned} \tag{3.3}$$

Where L_e and L_i are the emitted radiance in direction ω_o and the incident radiance in direction ω_i at x respectively. Ω represents the hemisphere or sphere covering all incoming directions at x. f_r is the *bidirectional reflectance distribution function* (BRDF) [33], while $\cos\theta_i$ is the inner product between L_i and N_x that is the normal direction at x. This integral term $L_r(\mathrm{x}, \omega_o)$ cannot be directly used to estimate radiance through the way of gathering the power of photons stored in the photon maps. Therefore, we must rewrite this integral to make it possible to deal with the flux (Φ). Firstly, $L_i(\mathrm{x}, \omega_i)$ can be converted into the formula below:

$$L_i(\mathrm{x}, \omega_i) = \frac{d\Phi_i(\mathrm{x}, \omega_i)}{d\omega_i dA_i^{\perp}} = \frac{d\Phi_i(\mathrm{x}, \omega_i)}{d\omega_i dA_i \cos\theta_i} \tag{3.4}$$

Where $d\Phi_i$ is the incoming flux at x, and $dA_i^{\perp}$ is the vertical differential area that is struck by incident direction ω_i. We then substitute $L_i(\mathrm{x}, \omega_i)$ of Equation 3.3 with Equation 3.4 and rewrite it as the following equation:

$$\begin{aligned} L_o(\mathrm{x}, \omega_o) &= L_e(\mathrm{x}, \omega_o) + \int_\Omega f_r(\mathrm{x}, \omega_i, \omega_o)\, L_i(\mathrm{x}, \omega_i) |\cos\theta_i| d\omega_i \\ &= L_e(\mathrm{x}, \omega_o) + \int_\Omega f_r(\mathrm{x}, \omega_i, \omega_o) \frac{d\Phi_i(\mathrm{x},\omega_i)}{d\omega_i dA_i \cos\theta_i} |\cos\theta_i| d\omega_i \\ &= L_e(\mathrm{x}, \omega_o) + \int_\Omega f_r(\mathrm{x}, \omega_i, \omega_o) \frac{d\Phi_i(\mathrm{x},\omega_i)}{dA_i}. \end{aligned}$$

(3.5)

Jensen adopted the nearest-neighbor density estimation method [28] to approximate incoming flux Φ_i. This method is to locate *N* photons from the photon maps which are closest to query point x, and accumulate the total flux of the *N* photons. Based on the assumption that all of the *N* photons and the surface intersect at x, hence Equation 3.5 can be rewrite again as follows:

$$\begin{aligned} L_o(\mathrm{x}, \omega_o) &= L_e(\mathrm{x}, \omega_o) + \int_\Omega f_r(\mathrm{x}, \omega_i, \omega_o) \frac{d\Phi_i(\mathrm{x},\omega_i)}{dA_i} \\ &\approx L_e(\mathrm{x}, \omega_o) + \sum_{p=1}^{\mathrm{N}} f_r(\mathrm{x}, \omega_p, \omega_o) \frac{\Delta\Phi_p(\mathrm{x},\omega_p)}{\Delta\mathrm{A}} \\ &\approx L_e(\mathrm{x}, \omega_o) + \frac{1}{\pi\mathrm{r}^2} \sum_{p=1}^{\mathrm{N}} f_r(\mathrm{x}, \omega_p, \omega_o) \Delta\Phi_p(\mathrm{x}, \omega_p) \end{aligned}$$

(3.6)

Where $\Delta\Phi_p$ represents each photon's power and $\Delta\mathrm{A}$ is the volume of a sphere around x, that is expended to contain *N*-nearest neighbor photons such as shown in Figure 3.5. However, the *N* photons are assumed to reside in the same flat surface with x. Therefore, we can be more correct to estimate $\Delta\mathrm{A}$ by computing the circle surface area from the sphere-surface intersection. That is, $\Delta\mathrm{A}$ equals to $\pi\mathrm{r}^2$ where r is the distance between x and the furthest photon.

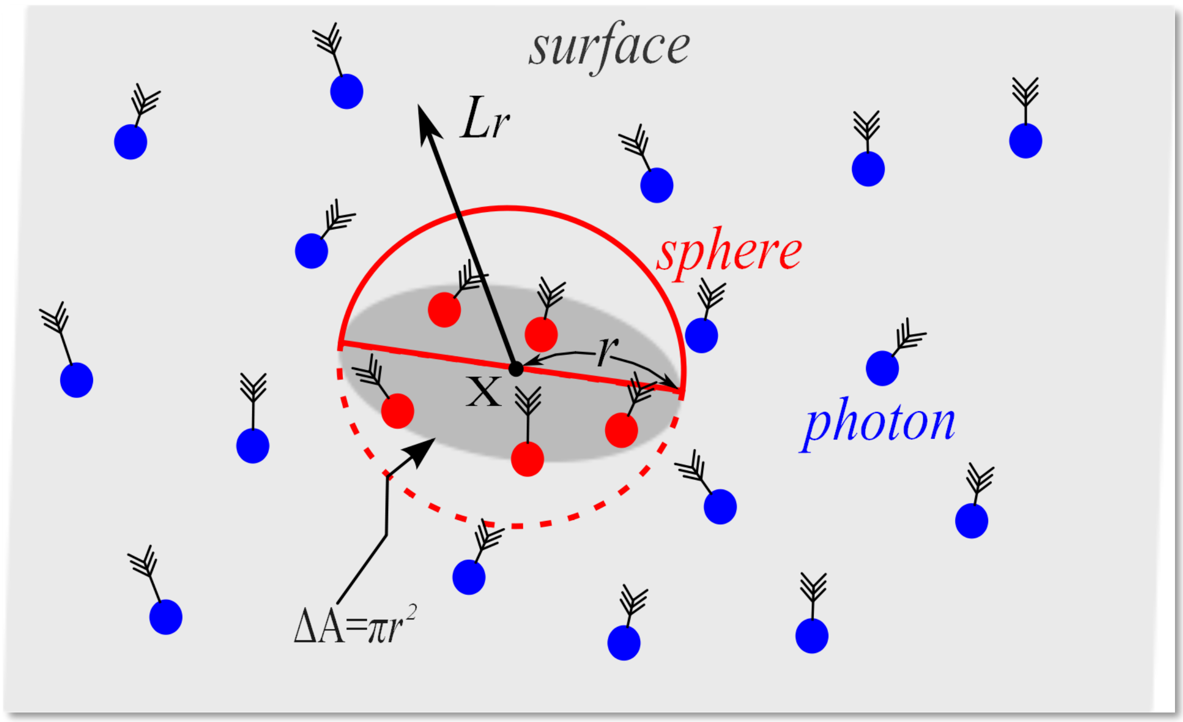

Figure 3.5. The reflected radiance L_r is estimated by using an expandable sphere to locate the nearest-neighbor photons around x and computing the disk area ΔA from the sphere-surface intersection.

3.1.7. Rendering

In the rendering pass, *photon mapping* exploits *distributed ray tracing* [35] to render the final image. Each pixel's radiance on the image is computed by shooting multiple rays from the viewer through the pixel into the scene, and averaging the radiance results returned by the rays. The radiance estimate is executed on the surface hit by a ray. It is then computed using *the rendering equation* (as showed in Equation 3.6) to locate the N nearest- neighbor photons from the photon maps around the query point.

Based on the both considerations of efficiency and quality on the radiance estimate, *the rendering equation* (Equation 3.3) is decomposed into four individual terms: direct illumination ($L_{i,l}$), specular reflection, caustics ($L_{i,c}$), and soft indirect illumination ($L_{i,d}$). The terms can be flexibly evaluated by utilizing different methods to visualize direct and indirect illumination as described in the next two paragraphs. The four equation terms (L_e

is omitted for convenience) are showed below:

$$\begin{aligned} L_r(\mathrm{x},\omega_o) &= \int_\Omega f_r(\mathrm{x},\omega_i,\omega_o)\, L_i(\mathrm{x},\omega_i)|\cos\theta_i|d\omega_i \\ &= \int_\Omega f_r(\mathrm{x},\omega_i,\omega_o)\, L_{i,l}(\mathrm{x},\omega_i)|\cos\theta_i|d\omega_i + \\ &\int_\Omega f_{r,s}(\mathrm{x},\omega_i,\omega_o)\, \left[L_{i,c}(\mathrm{x},\omega_i)+L_{i,d}(\mathrm{x},\omega_i)\right]|\cos\theta_i|d\omega_i + \\ &\int_\Omega f_{r,d}(\mathrm{x},\omega_i,\omega_o)\, L_{i,c}(\mathrm{x},\omega_i)|\cos\theta_i|d\omega_i + \\ &\int_\Omega f_{r,d}(\mathrm{x},\omega_i,\omega_o)\, L_{i,d}(\mathrm{x},\omega_i)|\cos\theta_i|d\omega_i \end{aligned} \tag{3.7}$$

Where,

$f_r(\mathrm{x},\omega_i,\omega_o) = f_{r,s}(\mathrm{x},\omega_i,\omega_o) + f_{r,d}(\mathrm{x},\omega_i,\omega_o)$, and

$L_i(\mathrm{x},\omega_i) = L_{i,l}(\mathrm{x},\omega_i)+L_{i,c}(\mathrm{x},\omega_i)+L_{i,d}(\mathrm{x},\omega_i)$

$L_{i,l}$ represents the direct illumination contribution from light sources, $L_{i,c}$ is caustic illumination while light sources has first performed specular reflection or transmission at least once, and $L_{i,d}$ is indirect soft illumination while light sources has been diffusely reflected at least once. The BRDF f_r is also divided into two components: $f_{r,s}$ and $f_{r,d}$. $f_{r,s}$ represents the specular component that is applied to highly glossy or perfect specular surface, while $f_{r,d}$ is the diffuse component that deals with perfect diffuse through slightly glossy surface.

3.1.8. Direct Illumination

There are three different ways to speed up computing direct illumination. The first method is a fast and approximate solution by locating direct and shadow photons (carry negative power) from the global photon map and directly substituting them into Equation 3.6 to estimate radiance.

The second method is similar to the above-mentioned way but evaluating the query point's visibility to light sources as shown in Equation 3.11. Then the radiance estimate can be obtained through directly multiplying V_l by the irradiance of light sources *l*.

$$V_l = \frac{n_{l,d}}{n_{l,d}+n_{l,s}}, \tag{3.8}$$

Where $n_{l,d}$ and $n_{l,s}$ are the number of direct and shadow photons respectively. V_l is the visibility rate to the light sources *l*.

In comparison to the first two ways, the final method is a more accurate estimate by combining V_l with shadow rays to detect the visibility to light sources. If V_l equals to zero or one, we can directly evaluate the result without wasting time to trace shadow rays. Otherwise, when direct and shadow photons coexist in the nearest-neighbor photons, this means that the query point might be within a penumbra region. At this moment, some shadow sample rays will be spawned at the query point and traced through the scene to test whether there is any obstacle occluding light sources *l*.

3.1.9. Specular Reflection

The second term of Equation 3.7 is to estimate the reflected radiance on highly glossy or specular surface. Because the outgoing directions on the specualr surface are almost reflected along the mirror direction, we barely observer any ray at the other directions. (see Figure 3.6) This means that all or most of the photons located from the photon map may not contribute to the radiance estimate. As a result, it leads to an incorrect result. So, the integral term is using standard Monte Carlo *ray tracing* method to spawn a few sample rays according to the *importance sampling* on BRDF $f_{r,s}(\mathrm{x}, \omega_i, \omega_o)$. These rays are recursively traced through the surfaces until they are terminated or reflected off a non-specualr surface and then return the estimates.

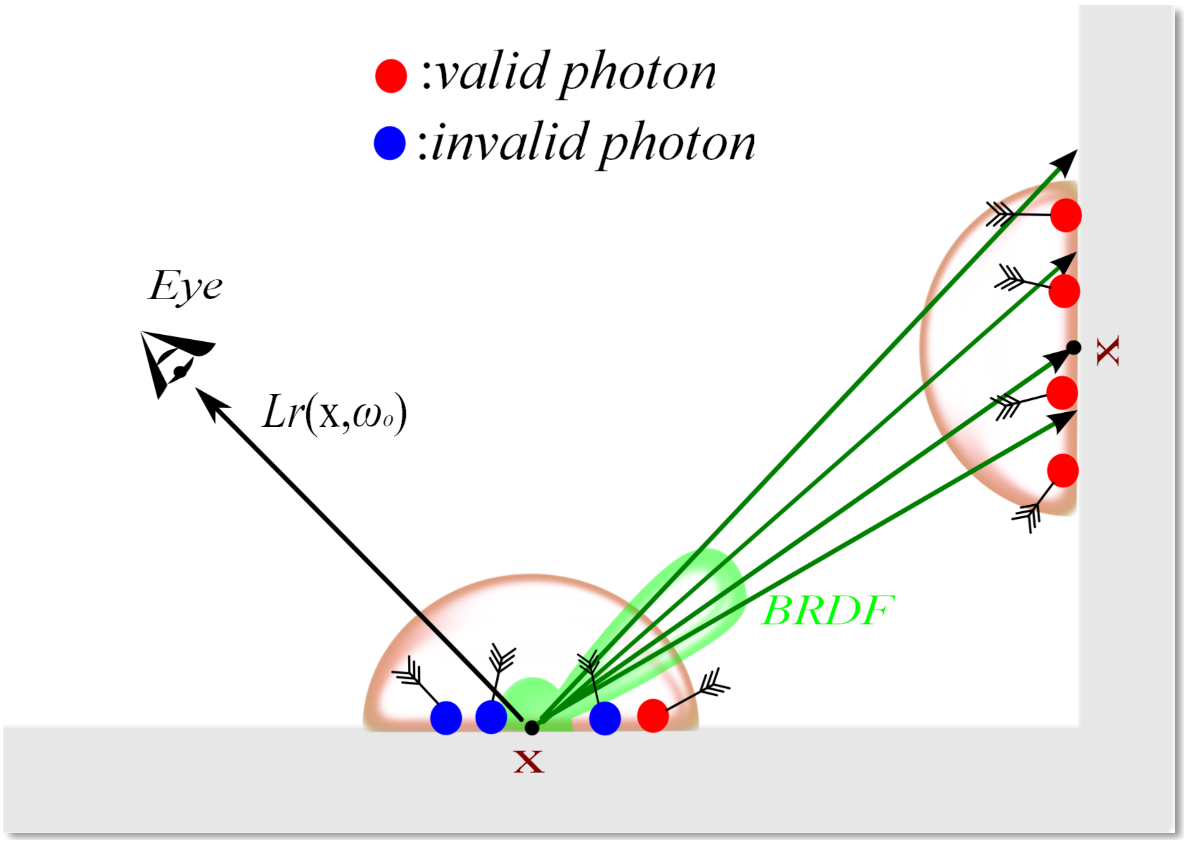

Figure 3.6. Specular reflection. It is not feasible to search photons on specualr surfaces which are stored in the photon map.

3.1.10. Caustic Illumination

Caustics are generated on diffuse or slightly glossy surfaces. They are evaluated by the third integral term of Equation 3.7. *Photon mapping* stores all of caustic photons in the both global and caustic photon maps. We can choose either one of them to directly visualize caustic illumination. The estimate difference appears on the types of the nearest photons located by the query point. That is to say, using the global map makes the *N*-closest photons possible to consist of direct, indirect, or caustic photons, but the photons only consist of caustic photons while using caustic map. Therefore, the latter gives a more accurate solution than the former because of locating more caustic photons. In addition, since caustic illumination is high variance, the photon map needs storing a large number of caustic photons to obtain a better image quality.

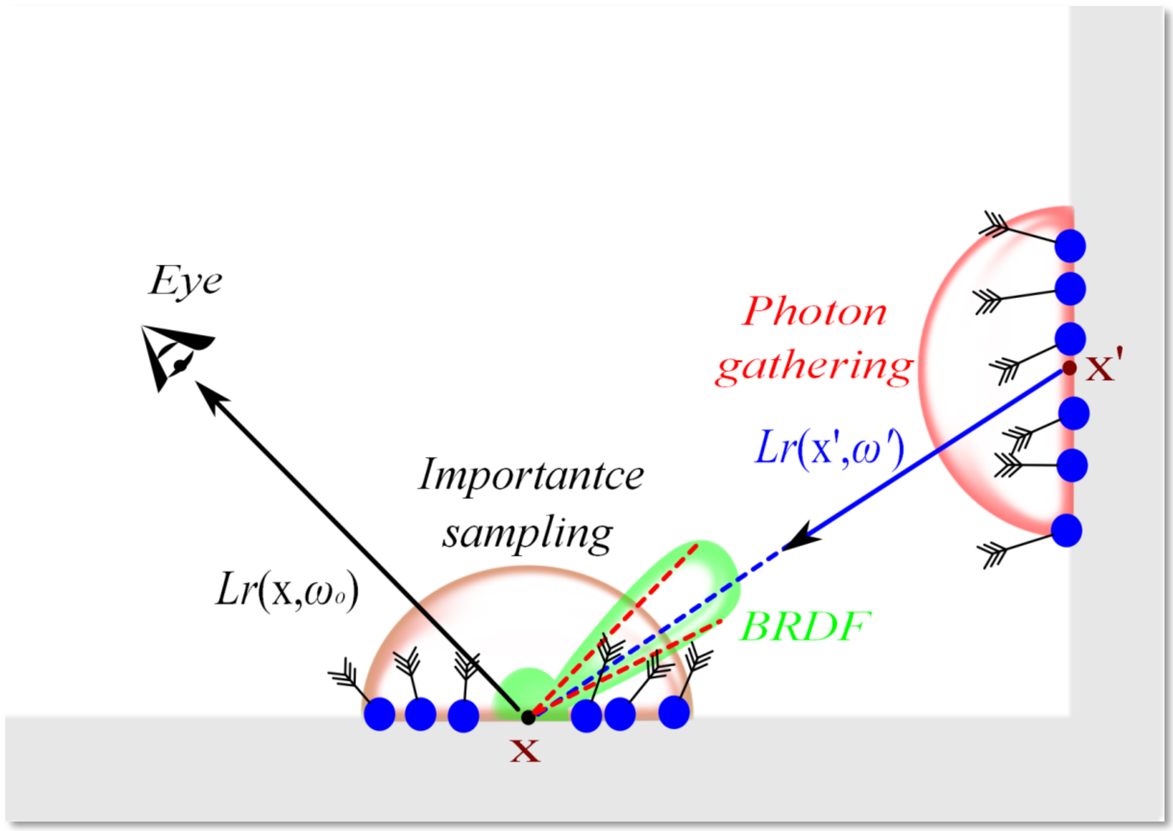

Figure 3.7. The *final gathering* technique is to trace a number of rays decided by *importance sampling* based on the BRDF distribution and the incoming flux at query point x to find the gather point x', and estimate $L_r(x, \omega_o)$ by gathering the reflected radiance $L_r(x', \omega')$ from all of the sample rays.

3.1.11. Soft Indirect Illumination

The last term of Equation 3.7 is applied to visualize soft indirect illumination on diffuse surfaces. It is computed by evaluating the incoming radiance which has been reflected off any diffuse surface at least once. We can know that the contribution to the reflected radiance L_r is so small that the image result looks smooth and soft. This effect is often called *color bleeding*.

There are mainly two methods to visualize soft indirect illumination. A fast approximate solution is directly using the global photon map to estimate radiance. If we need an accurate evaluation to obtain high quality, this can be achieved by the *final* *gathering* technique [35]. It adopted the Monte Carlo *ray tracing* method which uses an optimal *importance sampling* scheme based on both the BRDF and the incoming flux from the global photon map. That means that the optimal probability distribution function $p(\mathrm{x}, \omega_i)$ at query point x is decided by the following equation:

$$\begin{aligned} p(\mathrm{x}, \omega_i) &\propto f_r(\mathrm{x}, \omega_i, \omega_o) L_i(\mathrm{x}, \omega_i) \cos\theta_i \\ &= f_r(\mathrm{x}, \omega_i, \omega_o) \frac{d^2\Phi(\mathrm{x}, \omega_i)}{dA d\omega_i} \end{aligned} \tag{3.9}$$

In order to estimate the reflected radiance $L_r(\mathrm{x}, \omega_o)$ at query point x, as showed in Figure 3.7 above, we first use optimal *importance sampling* to spawn a limited number of sample rays and directions at x. Each ray is then traced through the scene to find next intersection point x' i.e. a gathering location. Finally, all of the rays return their reflected radiance $L_r(\mathrm{x}', \omega')$, which is computed at the gathering point by directly using the global photon map, to estimate the radiance $L_r(\mathrm{x}, \omega_o)$.

3.2. Level of Detail

3.2.1. Introduction

In *computer graphics*, *level of detail* (LOD) [36] is a technique that manages the complexity of 3D model representation by some metrics such as the object distance from the viewer, the geometry importance, or the view dependence. Figure 3.8 illustrates a bunny example based on the object distance from the viewer. When the bunny's distance is farer and farer, it is sufficient to represent the 3D object with less and less detailed representations. This technique increases the efficiency of rendering the scene and maintains a quality image concurrently.

Basically, there are three major categories for managing *level of detail*: *discrete*, *continuous*, and *view-dependent* LOD [37].

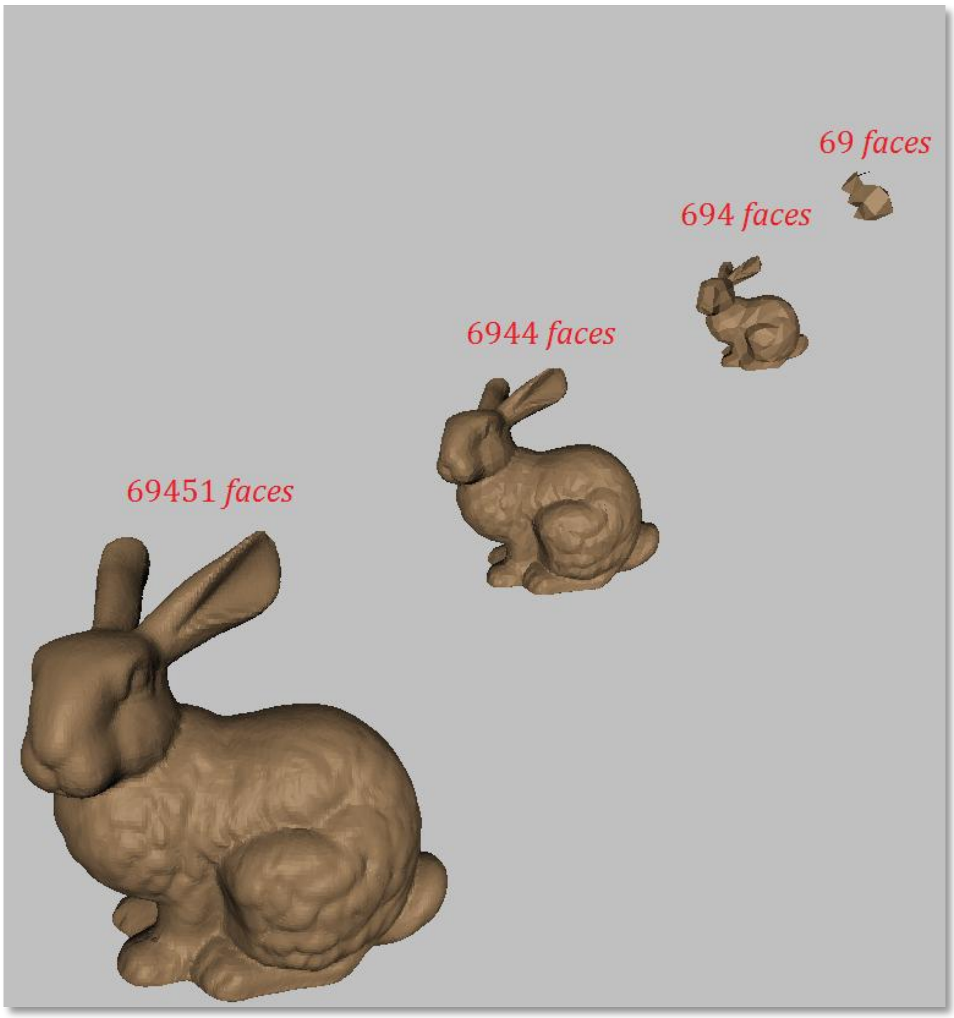

Figure 3.8. The bunnies are represented by different versions of *level of detail* based on the distance from the viewer.

Discrete LOD is a traditional approach originating from Clark [36] in 1976. At the preprocess stage, this method has first generated a finite number of *level-of-detail* versions for an object model. Then, at the run-time stage, the renderer just needs to decide a suitable detailed object version to be rendered in the scene. Because *discrete* LOD does not consider which the parts of an object the viewer can see to adaptively simplify the local geometric detail, it is also called *view-independent* LOD. A major advantage is that the simplification stage is isolated from the rendering stage so that the programming complexity becomes relative low. Therefore, the simplification stage has a large flexibility to create any number of *level-of-detail* models, while the render is simply responsible to choice which one detailed model to be rendered. However, in contrast, it needs to allocate a large number of resources for storing different versions of models. In addition, the render is forced to pick either an insufficient or excessive detailed model in an intermediate level, when we cannot find any appropriate detailed model to be rendered. In which case, the lower detailed version would affect the image quality while the higher detailed version needs more rendering time.

Compared to *discrete* LOD generating individual detailed objects in the preprocess stage, *continuous* LOD enables the renderer to dynamically render an appropriate detailed version at run time. This method first extracts the simplification information by creating a data structure to evaluate the error metric at the simplification stage. At run time, the renderer then dynamically determines an optimal version using the error metric according to the desirable simplification demand. This technique has a better granularity so that we may make use of the resources to generate a high quality image.

View-dependent LOD is an extension of *continuous* LOD. Depending on view-dependent simplification metric, the system dynamically determines an optimal *level-of-detail* version for different viewpoints. A great advantage is that the different parts of an object model can support multiple levels of simplification. For example, a visible area near the viewer would show a higher detail than the other parts. This approach brings an even better granularity than the above-mentioned methods, especially for some large objects such as terrains.

3.2.2. Polygonal Simplification using Quadric Error Metrics

Garland and Heckbert [24] proposed a surface simplification algorithm using *quadric error metrics* (QEM) which is a *continuous* LOD. The aim is achieving a trade-off between efficiency and quality by combining speedy simplification, and approximation fidelity, with approach generality. Based on iterative contraction of vertex-pairs, the algorithm performs a series of mesh simplification operations by maintaining a geometric error approximation using quadric matrices.

Initially, the algorithm selects all of candidate vertex-pairs, regardless of whether a pair shares an edge or not, and computes the quadric error for each pair to be as the contraction cost. The candidate vertex-pairs are then placed in a heap queue to sort according to the contraction-cost order. In each time of contraction, we merge a vertex-pair with the minimum cost and re-evaluate the quadric errors for the associated pairs which connect with the removed vertex-pair. Finally, the simplification step is iteratively performed, until the desired number of polygons or approximation has been reached.

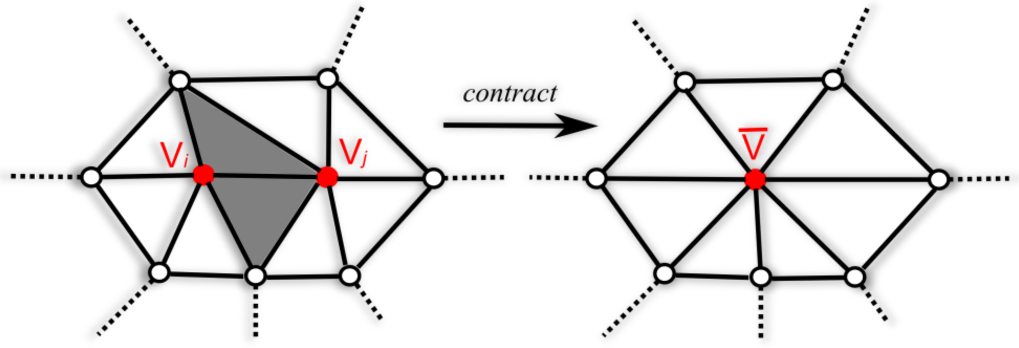

Figure 3.9. The process for contracting a vertex-pair. A candidate vertex-pair (v_i, v_j) is selected, and collapsed into a new target vertex $\bar{v}$. Both grey regions related to (v_i, v_j) are then degenerated and removed.

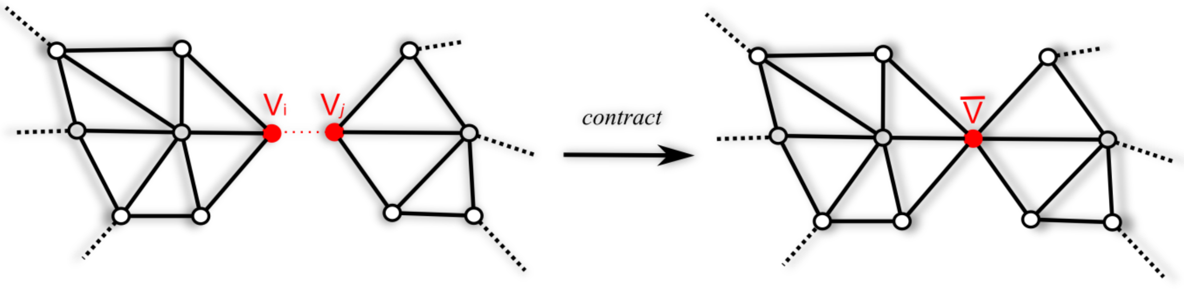

Figure 3.10. An unconnected vertex-pair (v_i, v_j) is selected as a candidate, where v_i and v_j come from two separate regions. The pair is contracted into a new vertex $\bar{v}$ such that both regions are merged into a non-manifold surface.

3.2.3. Candidate Vertex Pairs

QEM algorithm adopted a vertex-pair collapse operator to make it possible to topologically simplify manifold and non-manifold surfaces. This can be done by supporting two contraction types including connected and unconnected vertex-pairs as showed in Figure 3.9 and Figure 3.10 respectively. As mentioned above, a candidate vertex-pair (v_i, v_j) must correspond to either of the following conditions:

a. (v_i, v_j) has an edge

b. (v_i, v_j) has not an edge, and $\|v_i - v_j\| < d$,

Where the distance *d* is a threshold parameter specified by the user. When *d* equals to zero, the vertex-pair contraction will become a general edge-collapse operation. In contrast, higher threshold will result in a large number of unconnected vertex-pairs to be selected such that some undesirable results are generated. In fact, a smaller threshold is enough to be used in most cases.

3.2.4. Quadric Error Metrics

Based on the fidelity metric for managing the mesh-simplification process, LOD algorithms are generally divided into two types: *fidelity-based simplification* and *budget-based simplification* [37].

Fidelity-based simplification is controlled by assessing a simplification error ϵ, which represents the difference between the simplified mesh and the original mesh. This method can often bring an accurate approximation. There are some common error metrics for evaluating simplification error, including geometry error, attribute error, combining error, incremental error, and so on.

Budget-based simplification is managed by achieving the desirable number of polygon or vertex simplification which is specified by the user. Relatively, this method can increase the efficiency of the rendering pass.

Basically, *quadric error metrics* is geometry error metrics, but can be extended to become one kind of combing error metrics by incorporating some vertex attributes such as colors, textures, and surface normals [38]. In this dissertation, we only use a basic QEM algorithm, and therefore we do not describe the extension method for convenience.

In order to characterize the geometry error, QEM algorithm defined an error at a given vertex v to be the sum of squared distances from vertex v to all the associated planes. In which case, the planes are the triangles which intersect at vertex v. This error (Δv) at vertex v is represented as follows:

$$\Delta \mathrm{v} = \Delta\left(\left[\mathrm{V_x}, \mathrm{V_y}, \mathrm{V_z}, 1\right]^{\mathrm{T}}\right) = \sum_{\mathrm{p} \in \mathrm{planes}\ (\mathrm{v})} (\mathrm{p^T v})^2 \tag{3.10}$$

Where the 3D coordinates of vertex v is denoted as a transpose matrix. p equals to $[\mathrm{a\ b\ c\ d}]^{\mathrm{T}}$ where a, b, and c are the coefficient term, and d is the constant term of the plane equation $\mathrm{ax} + \mathrm{by} + \mathrm{cz} + \mathrm{d} = 0$. Also, the equation has been normalized such that $\mathrm{a}^2 + \mathrm{b}^2 + \mathrm{c}^2 = 1$. Furthermore, the error metric can be rewritten as the quadratic form below:

$$\begin{aligned} \Delta \mathrm{v} &= \sum_{\mathrm{p} \in \mathrm{planes}\ (\mathrm{v})} (\mathrm{v^T p})(\mathrm{p^T v}) \\ &= \sum_{\mathrm{p} \in \mathrm{planes}\ (\mathrm{v})} \mathrm{v^T}(\mathrm{pp^T})\mathrm{v} \\ &= \mathrm{v^T}\left(\sum_{\mathrm{p} \in \mathrm{planes}\ (\mathrm{v})} \mathrm{K_p}\right)\mathrm{v} = \mathrm{v^T Q v} \end{aligned} \tag{3.11}$$

Where we substitute coefficients a, b, c, and constant d into p and obtain the fundamental error quadric matrix $\mathrm{K_p}$ as below:

$$K_p = pp^T = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} [a\ b\ c\ d] = \begin{bmatrix} a^2 & ab & ac & ad \\ ab & b^2 & bc & bd \\ ac & bc & c^2 & cd \\ ad & bd & cd & d^2 \end{bmatrix}$$

(3.12)

As showed in Equation (3.11), we know that the error at any vertex in space can be efficiently computed by first summing all of error quadric matrices to become a single matrix Q. A more important meaning is that while contracting a vertex-pair (v_i, v_j) into a new position $\bar{v}$, the quadric matrix $\bar{Q}$ for vertex $\bar{v}$ can be computed by solving the problem of the union set $(\text{planes}(v_i) \cup \text{planes}(v_j))$. If $\text{planes}(v_i)$ and $\text{planes}(v_j)$ are disjoint, we can directly add Q_i and Q_j together to obtain $\bar{Q}$. Otherwise, if they are overlap, this algorithm suggested that we can sacrifice some quality to efficiently adopt the addition operation $(Q_i + Q_j)$, or additional storage to solve inclusion-exclusion formula on demand.

The only remaining problem is how to determine a position for vertex $\bar{v}$ with minimum error $\Delta\bar{v}$. QEM offers a simple solution which picks a position from either of v_i,v_j, or $(v_i + v_j)/2$ to compute $\Delta\bar{v}$. The position with the smallest error value will be chosen as the new vertex $\bar{v}$.

Another method is to attempt to find a minimum value from the partial differential equation, i.e. $\frac{\partial\Delta\bar{v}}{\partial x} = \frac{\partial\Delta\bar{v}}{\partial y} = \frac{\partial\Delta\bar{v}}{\partial z} = 0$, by expanding $\Delta\bar{v}$ to become a quadric polynomial equation as showed in Equation (3.13):

$$\Delta\bar{v} = \bar{v}^T Q \bar{v}$$

$$= [x\ y\ z\ 1] \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ q_{14} & q_{24} & q_{34} & q_{44} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$= q_{11}x^2 + q_{22}y^2 + q_{33}z^2 + 2q_{12}xy + 2q_{13}xz + 2q_{14}x + 2q_{23}yz + 2q_{24}y + 2q_{34}z + q_{44}$$

(3.13)

Where we assume that $\bar{v} = [x\ y\ z\ 1]^T$ and q_{kl} denotes the element in row k and column l. In addition, this partial differential equation can be further represented as a matrix form to find the solution:

$$\begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \bar{v} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

(3.14)

If this matrix is invertible, we can get the following solution:

$$\bar{v} = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

(3.15)

Otherwise, if it is not invertible, the only way is to extend the first method to find an approximate position with a relative smaller $\Delta\bar{v}$ from the interval between v_i, and v_j.

3.3. Diamond Optics

Diamond's preciousness not only denotes its rareness, but also its fascinating appearance. That is to say, diamond may exhibit more brilliant and colorful visual effects than the other objects. This is because it has a perfect cut and polished appearance such as Figure 3.11, and shows complicated optical properties such as high reflectance, high refractive index, noticeable dispersion, intense caustics, and polarization. Polarization will be not discussed and implemented for efficiency.

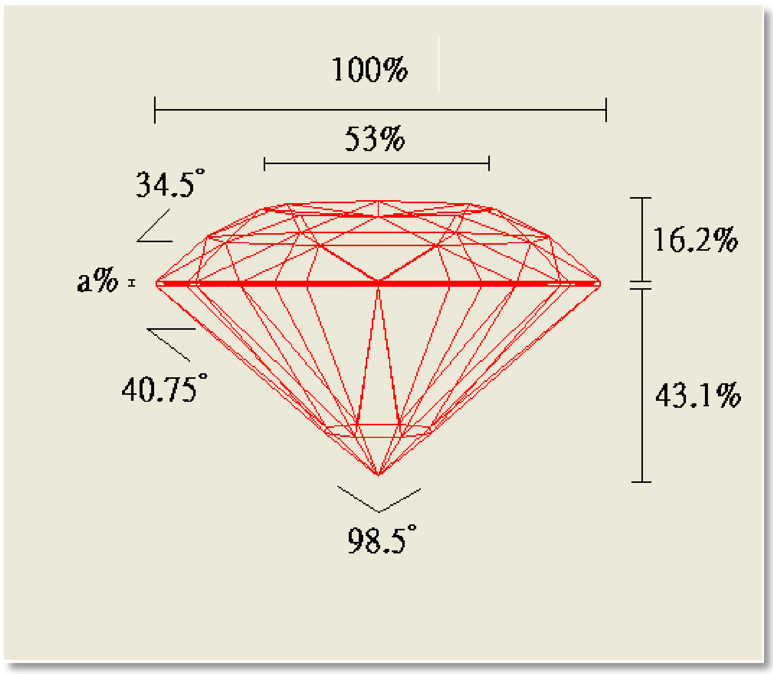

Figure 3.11. The cutting rate of a round brilliant cut.

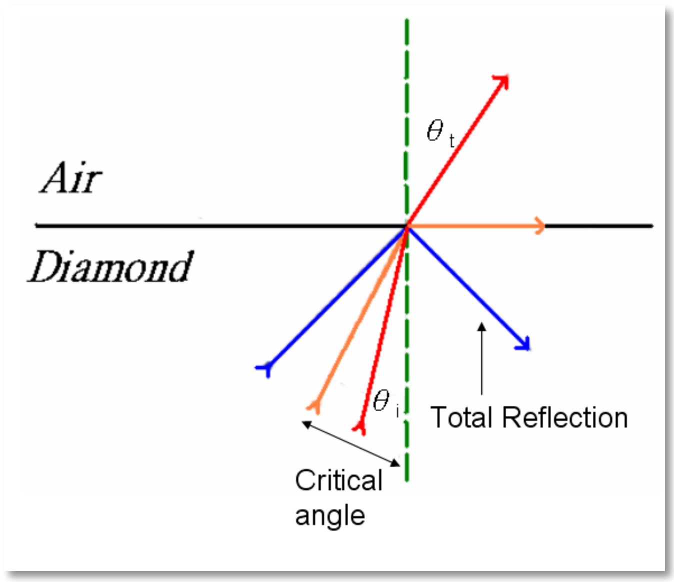

Figure 3.12. Total reflection happens at the angle which is equivalent to or bigger than the critical angle where θ_i and θ_t are the incident angle and refractive angle respectively.

3.3.1. Total Reflection

When light strikes the inner surfaces of a well-polished diamond, in a general case some of light is reflected off the surfaces and the remaining light is transmitted into the outside of the object. However, if light strikes at the angle which is equivalent to or bigger than the critical angle (about 24.5 degrees), 100% energy of light would be totally reflected as showed in Figure 3.12.

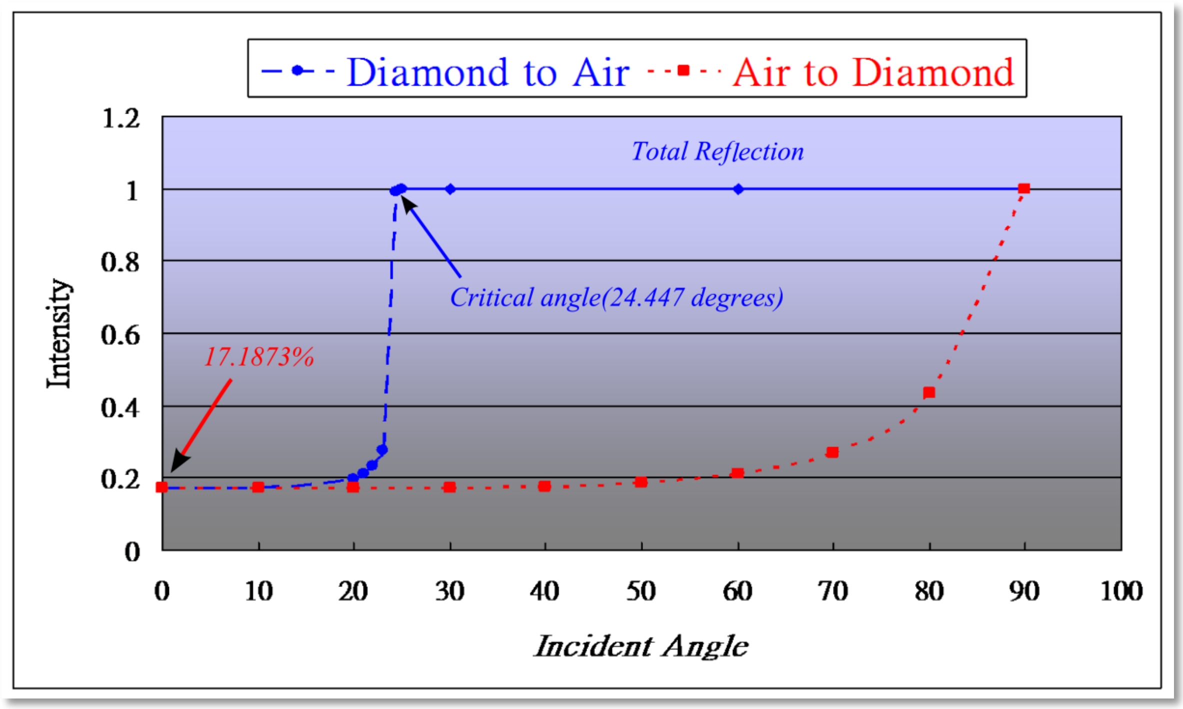

Figure 3.13. Reflective intensity v.s. Incident angle.

We can observe that when incident light strikes the outer surfaces of diamond, the critical angle is the maximum refractive angle generated from whatever incident light enters at which angle. In addition, from *Snell's Law* ($n_i \sin\theta_i = n_t \sin\theta_t$), we can also know that the higher a medium's refractive index has, the smaller critical angle it generates. Because diamond has a very high refractive index, its critical angle becomes so small that light tends to be totally reflected. This is also the reason why a large amount of light can be transmitted through diamond into air. The above-mentioned trend can be illustrated as Figure 3.13. With the aid of *Fresnel formula*, this graph records how much of energy is reflected or refracted at different incident angles from air into diamond or diamond into air.

3.3.2. Dispersion and Fire

Conceptually, fire is not the same as dispersion. It can be observed from an appearance aspect, where we can see colored flashes. In contrast, dispersion is inherent to transparent materials. It originates from the fact that white light is broken into various monochromatic lights. In brief, fire is the resultant production after white light being dispersed into colored lights.

Several factors would affect either effect of dispersion or fire. First, both angle and position of incident or outgoing light would strongly impact the effect the viewer can see. Second, the longer distance a light beam propagates inside the diamond, the wider it spreads out. Finally, it is also affected by light spreading out across at least two adjacent facets, where a dispersed light-beam will split apart and propagate in different directions. (See Figure 3.14)

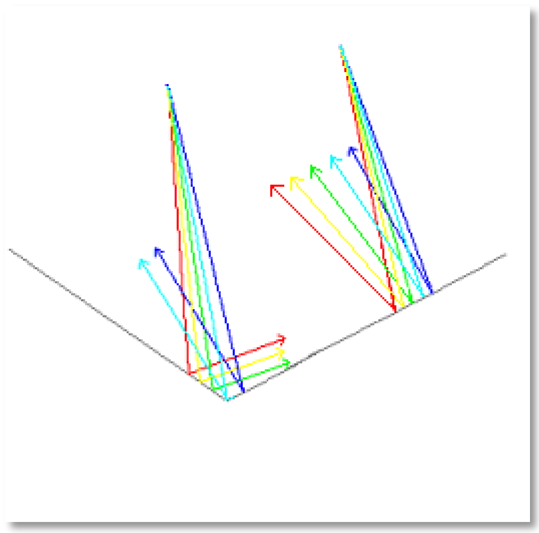

Figure 3.14. A dispersed light-beam splits apart and spreads out in different directions.

3.3.3. Geometry Cutting

Cutting a model into different geometric shapes, such as shown in Figure 3.15 and Figure 3.16, also influences the behavior of light propagating inside the diamond. For example, the brilliant-cut such as Figure 3.11 is designed to have most of the incident light be transmitted into air after performing multiple times of total reflections inside the diamond. In addition, diamond is made up of well-polished facets which are extremely similar to specular surfaces. It, therefore, has very high reflectance and noticeable caustic effects.

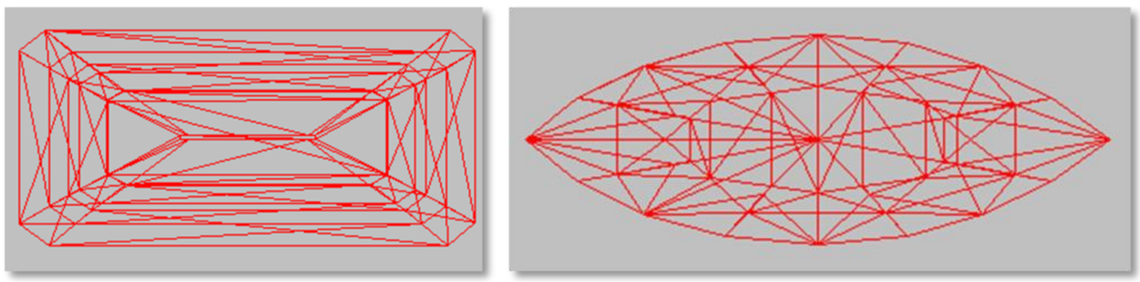

Figure 3.15. Emerald-cut and marquise-cut.

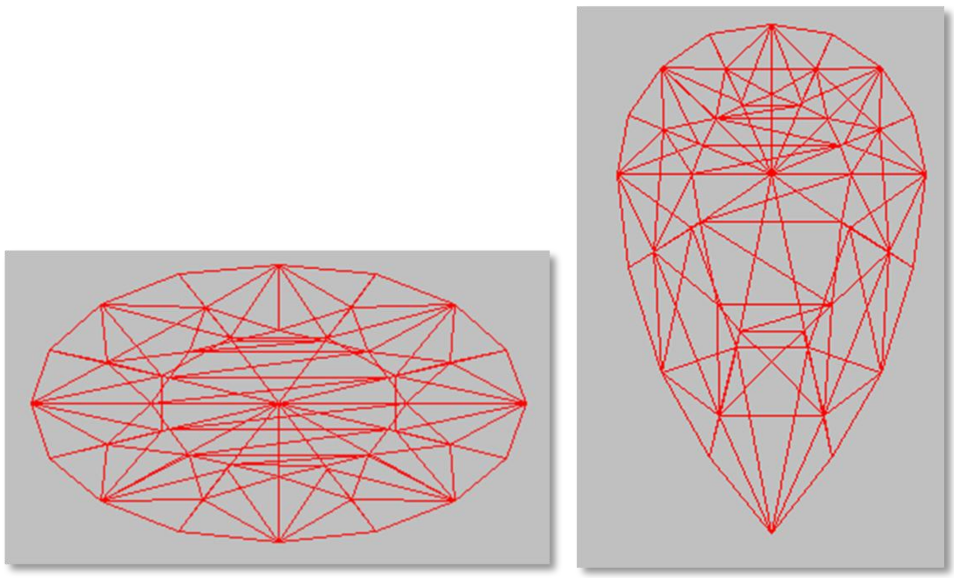

Figure 3.16. Oval-cut and pear-cut.

4. Grouped Photon Mapping

In this chapter, we detail the *grouped photon mapping* method. This chapter is organized into 4 sections. At the beginning, we explain the basic concept our algorithm adopts in Section 4.1, and give an overview of our algorithm in Section 4.2. In Section 4.3, a LOD technique, QEM, is used to manage the process of merging noise groups and obtain a balance between noise and bias. In Section 4.4, we maintain a polygonal boundary for each photon group to rebuild a beam-like illumination area. In Section 4.5, an advanced nearest-neighbor density estimation method with a double-layer kd-tree is proposed. Finally, we explain how to visualize global illumination by the GPM method in Section 4.6.

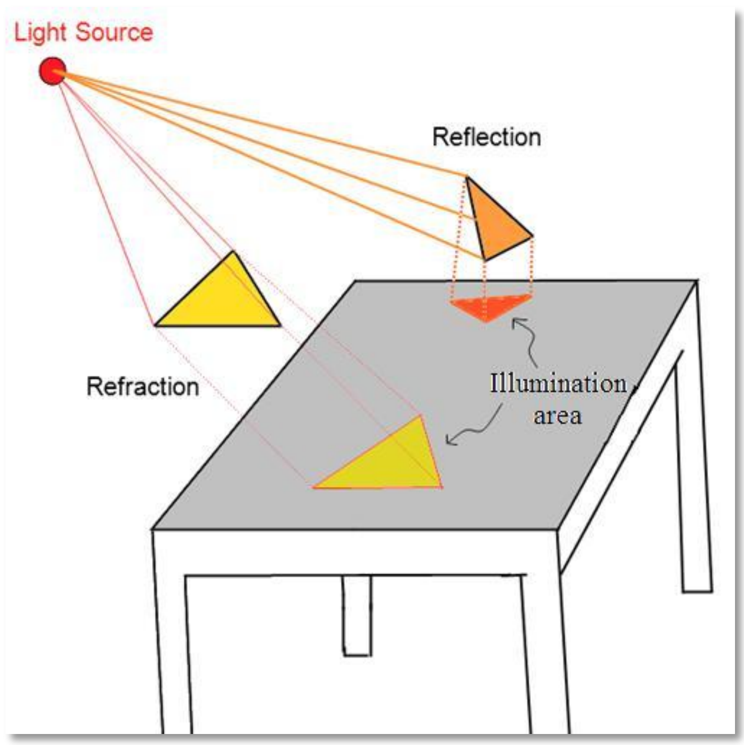

Figure 4.1. A reflected or refracted light-beam strikes the table and forms a visible illumination area.

4.1. Basic Concept

A light-beam can be represented as a bundle of individual rays that have the same *ray coherence* in their transmission paths where each one of the intersection, path, or direction corresponds to the coherence. In addition, when a light-beam is projected onto a surface, a clear and visible illumination area will be formed on this surface as shown in Figure 4.1 above. The illumination-area information is very useful to give an accurate reference while gathering any illumination contribution at a given point to render global illumination.

Based on the above light-beam concept, in this dissertation we combine the light-beam idea with the *photon mapping* method to rebuild different beam-like illumination areas using a large number of discrete photon-particles. In order to achieve the above-mentioned goal, we first record an *intersection history* for each traced photon in the photon-tracing pass as one kind of *ray coherence*. The *intersection history* is a collection of all the mesh indexes hit by a photon. According to the *intersection-history* coherence, we then cluster the photons together that have the same coherence to represent a unique light-beam. Furthermore, the illumination area can be rebuilt by the distribution of photons. Therefore, in the rendering pass, we can use the illumination areas to serve as a filter for managing the searchable area of a query point. It replaces an inaccurate searching area using the sphere or disc by standard *photon mapping*.

4.2. Overview

Grouped photon mapping is made up of the photon-tracing and rendering passes, as illustrated in Figure 4.2. Basically, our algorithm's most of stages follow the procedures of Jensen's method [10]. The first pass, photon pass, is *path tracing* which traces, clusters, and stores photons on diffuse surfaces by maintaining multiple independent grouped photon maps to store different *ray-coherence* groups of photons. Then we reconstruct the beam-like illumination areas by means of constructing polygonal boundaries. In the second pass or rendering pass, we adopt *distributed ray tracing* to render the final image result. We gather the energy of the photons sampled from the grouped photon maps to estimate the radiance for visualizing global illumination.

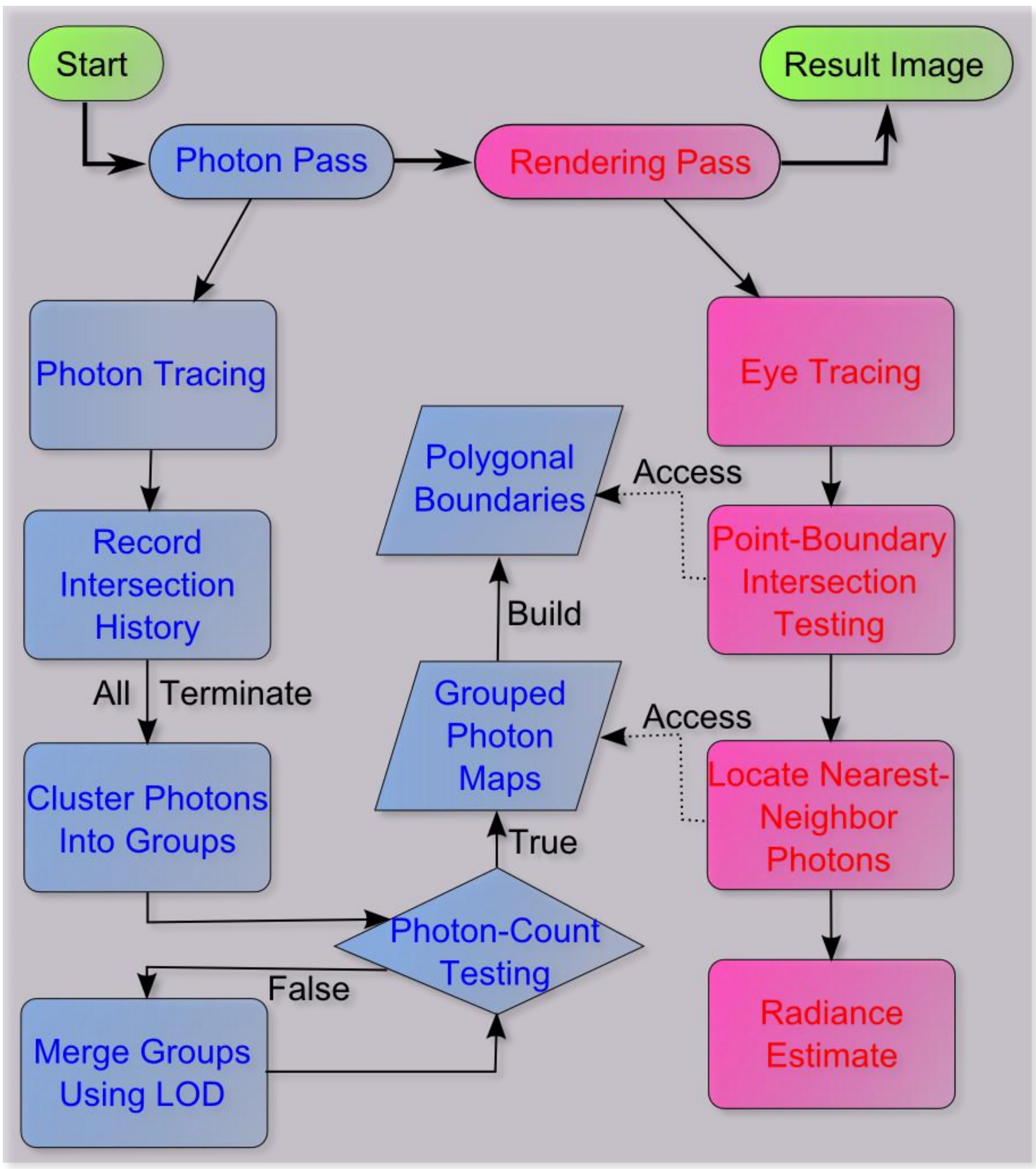

Figure 4.2. The system architecture for *grouped photon mapping*. It is separated into two main stages, including the photon and rendering passes. The photon pass performs seven procedures (blue-color blocks) to create multiple grouped photon maps, while the rendering pass takes four steps (pink-color blocks) to estimate the radiance for the final result.

4.2.1. Photon Pass

In this pass, initially, all photons are spawned from performing Monte Caro *uniform* on light sources as shown in Figure 4.3. The photons are then showered into the scene to perform a series of optical reactions with surfaces.

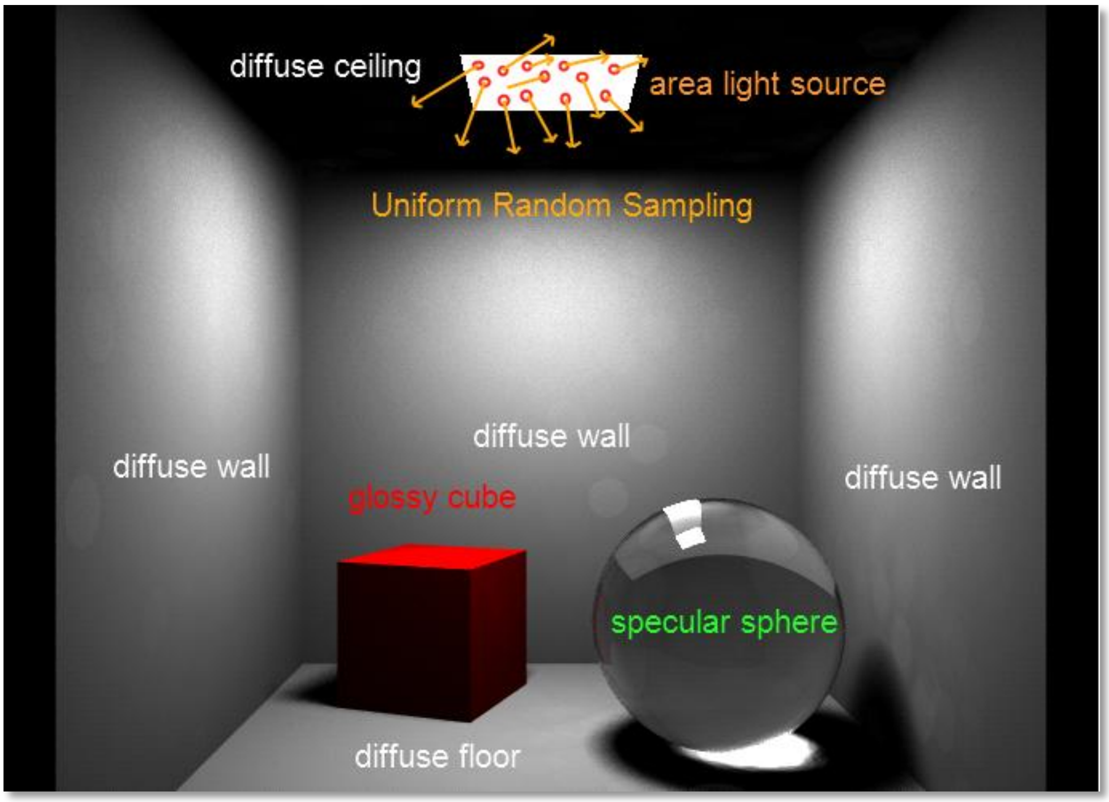

Figure 4.3. A photon is generated by performing *uniform random sampling* on the area light sources.

While a photon is being traced, we use a temporary array to record the *intersection history* as shown in Figure 4.4. This array stores all of the indexes of polygonal meshes hit by each ray-segment of a photon in the transmission path. In addition, in order to distinguish these photons which actually are spawned from different light sources, we also store the light source index in the head of the array.

Once all photons have terminated their transmissions, we then cluster them into different beam-like groups according to the intersection-history coherence. That is to say, when there are at least two photons sharing a total identical history, they are clustered together in a common group. These photons can compose a beam of light. (see Figure 4.5)

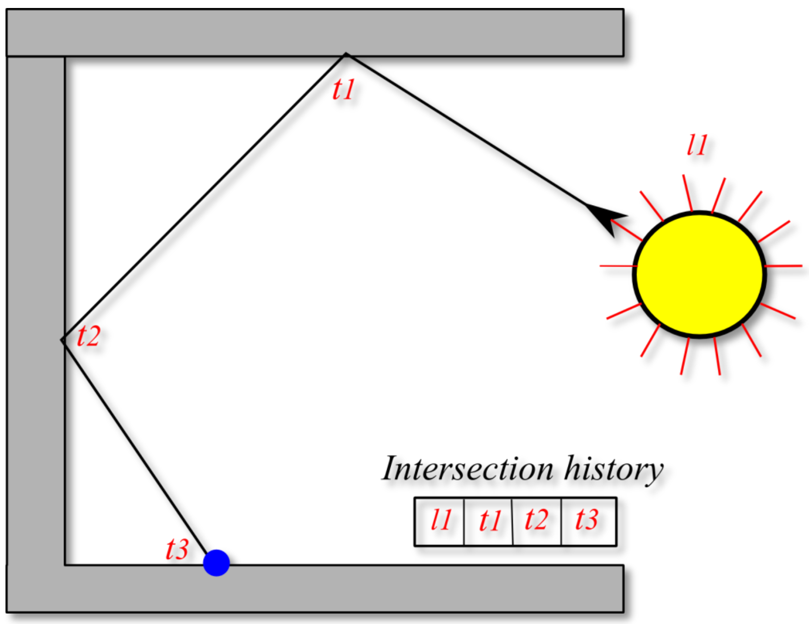

Figure 4.4. An intersection history is to record the light source index and all the indexes of meshes hit by a photon.

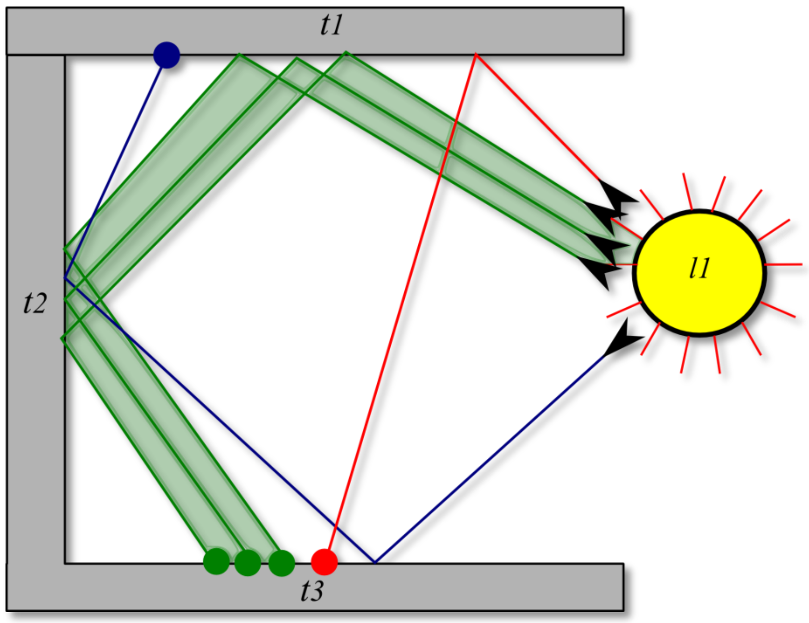

Figure 4.5. Three discrete photons (represented by green color rays) are clustered together in the same group based on the *intersection-history* coherence to rebuild a beam of light.

After finishing creating photon groups, we start to perform a photon-count testing for each group to detect whether the number of photons is insufficient or not. The reason for testing is that the insufficient groups would influence the reliability of rebuilding the beam-like illumination areas which depend on the distribution of photons in a group. These groups will make the result of the radiance estimate generate noise, so we call them noise groups. This problem usually becomes more obvious when a scene consists of models with large numbers of polygonal meshes. For the photons going through these models, they will be clustered into many more small groups with few photons because of generating more different intersection histories. Besides showering more photons to reduce this kind of error, we also adopt a *level of detail* (LOD) technique to indirectly merge those groups with few photons. The concept is that we iteratively merge adjacent groups which have similar intersection histories into a larger one, after each time of surface simplification. By this method, we can raise the photon-count of a group and eliminate noise groups. (This will be further explained in Section 4.3)

When terminating the photon-count testing for all groups, we separately create an independent photon map to store each photon group, instead of a standard photon map to store all photons. For each group, we also construct a unique kd-tree to organize and store its photon data and extend the *Graham's Scan* [40] method to maintain a polygonal boundary so as to reconstruct a beam-like illumination area. (This will be described in Section 4.4)

4.2.2. Rendering Pass

In the rendering pass, we exploit a new nearest-neighbor density estimation method. The sampling or searching area around a query point is no longer restricted or determined by the region of a predefined fixed-shape such as a disc or sphere. Instead, the sampling area relies on the intersection testing between the query point and photon groups to verify which group-areas can be queried. Only if the point intersects with the polygonal boundary of any one group, can we access its photon kd-tree to locate the nearest-neighbor photons. Until the *N* nearest-neighbor photons have been found, we continue performing the intersection testing with the other groups and doing the sampling operations.

However, it needs taking a great amount of time for a query point to perform the ray-boundary intersection testing with all groups. Therefore, we have built a double-layer kd-tree, which is composed of the group nodes at the first layer and the photon nodes at the second layer. A group node is represented by the center point of a bounding sphere which can include all the photons of a group. This double-layer kd-tree can first filter and eliminate most of unrelated groups prior to the ray-boundary intersection testing. (This will be detailed in Section 4.5)

In the final radiance estimate, we also use *Silverman kernel function* [14] to filter all of the photons sampled by the query point.

4.3. Merging Groups using Quadric Error Metrics

In the nearest-neighbor estimation method, we often encounter the problem that only few photons are sampled around the query point. This can lead to a visible error viewed as noise in the synthesized image. The common solution for standard *photon mapping* is increasing the bandwidth to search for more neighboring photons, or to directly ignore this result. However, relatively extending bandwidth also increases the scale of bias [11]. This problem is also reflected in our method. While creating photon groups, a group with few photons will result in a noisy result on the radiance estimate.

To achieve a trade-off between noise and bias, we introduce a LOD technique proposed by Garland and Heckbert [24]. They used an efficient *quadric error metrics* to evaluate the contraction cost of a vertex-pair. Each time they choose a candidate vertex-pair with the lowest cost to be contracted during the surface-simplification process. This ensures that there is a higher priority to eliminate the most unimportant meshes of a model so that we can keep the important features of the appearance. We make use of this advantage to indirectly determine the order of reorganizing noise groups which only include few photons, such as those with fewer than 3 photons. The basic merging concept is that we iteratively contract vertex-pairs to merge meshes and modify their identifiers (mesh index). Meanwhile, we utilize the modification information to update the photon's *intersection history*. By this way, we further merge the photons which have an identical *intersection history* into a big group, so that we can effectively raise the photon-count of a group and reduce the number of noise groups. The actual merging steps are as follows (see Figure 4.6 below):

a. First use QEM to determine a vertex-pair (V_i,V_j) with the lowest contraction cost.

b. Choose a vertex V_j with the lowest error ΔV_j to be removed and merged into V_i.

c. Locate the adjacent meshes (s1-s5 and s9) that connect with the removed vertex V_j.

d. Reassign these meshes (s1-s5 and s9) to the same index s1.

e. Detect which groups went through these merged meshes (s1-s5 and s9), and then change the relative mesh indexes of their *intersection histories* to be s1.

f. Finally, determine which the intersection histories of groups have been modified, and then merge the groups that have an identical intersection history into a larger one.

It is not necessary to reduce entire noise groups, because merging too many groups can greatly increase the bias. Thus, by a simple statistical and empirical method, we evaluate whether or not to terminate the merging action. In geometry the most basic polygonal shape is the triangle. We, therefore, can reasonably assume that a valid group which must include at least three photons is just sufficient to construct a beam-like illumination area. Otherwise, it would become a noise group. Based on the evaluation standard for a valid group, we can gather the statistics about the ratio of valid photons to total photons to determine when to stop the merging action. When a photon belongs to a valid group, we call it a valid photon. As a rule of thumb in our experiment, it has been quite sufficient to render a high-quality image when 90% of photons are merged into valid groups, as the results shown in Section 5.2.

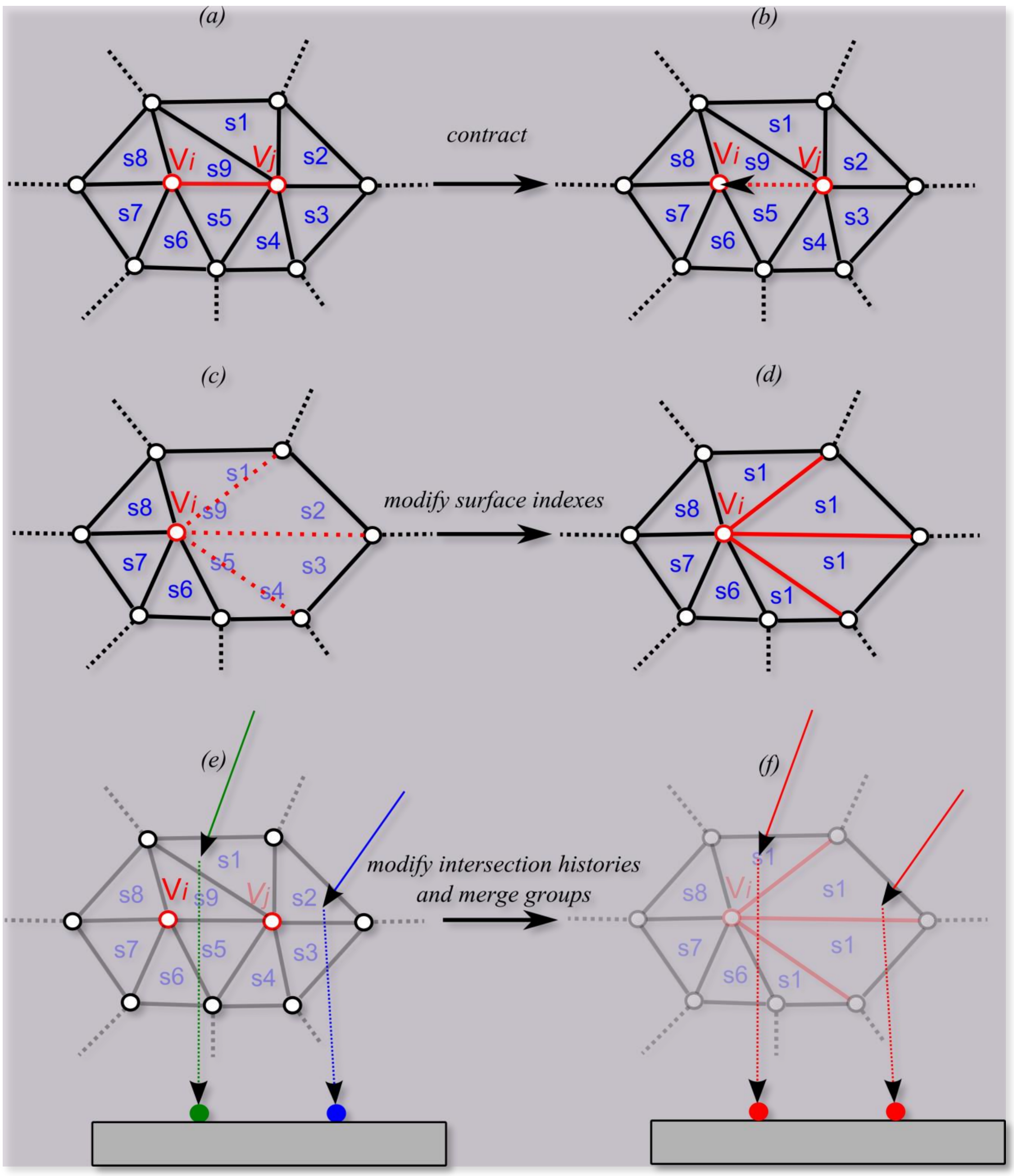

Figure 4.6. Using QEM to control the process of merging groups: (a) select a candidate vertex-pair (V_i,V_j) with the lowest contraction cost ($\Delta\overline{V}$), (b) merge V_j into V_i, (c) locate the adjacent meshes s1-s5 and s9, (d) reassign these mesh indexes to s1, (e) update the intersection histories for green and blue photons, (f) merge

photons with total identical history into a big group.

4.4. Polygonal Boundary

In general, constructing a 2-dimentional *convex hull* which includes all of the photons in a group is a convenient solution to approximate the illumination area of a light-beam. However, a beam could split into several sub-beams with irregular shapes when crossing a common border shared by multiple adjacent surfaces. To solve this problem, we have extended *Graham's Scan* algorithm [40]. Our extension uses 2-dimentional polygonal boundaries to create various shapes, including convex and concave forms. Figure 4.7 illustrates the steps used to construct the concave boundary for a photon group.

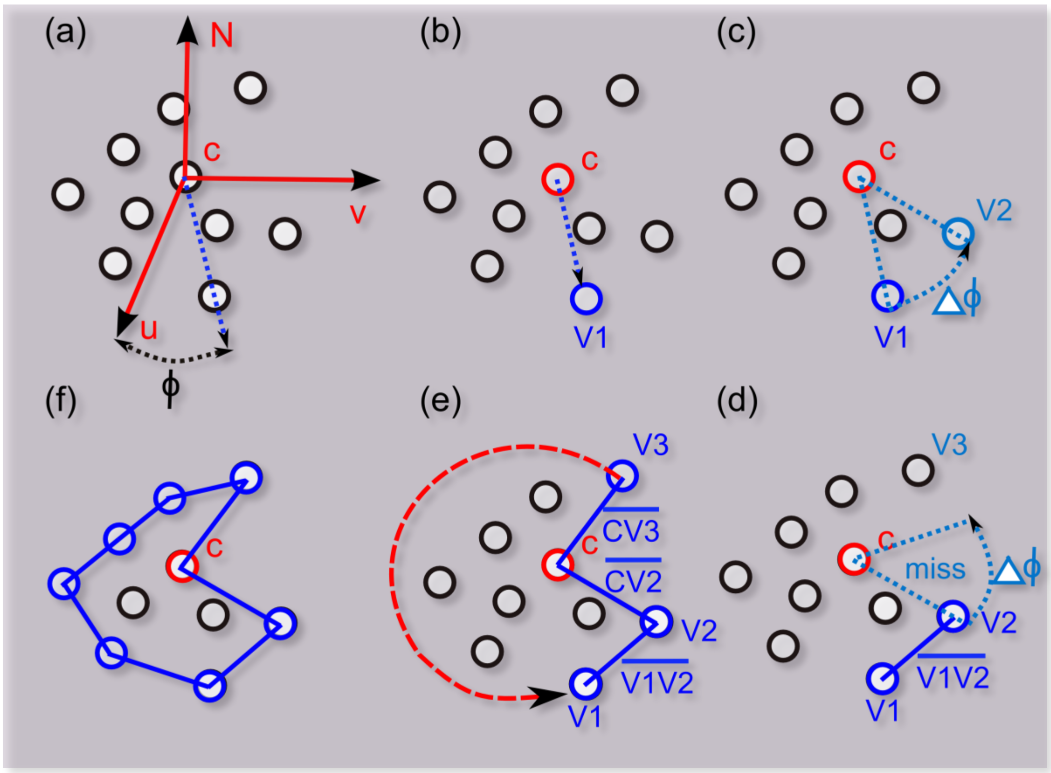

Figure 4.7. The processes of building a polygonal boundary: (a) find a pivot, C, which is closest to the core position, and sort the other photons by ϕ, (b) find the first vertex, V1, that is the farthest photon, (c)search for the next vertex, V2, within $\Delta\phi$, (d) create the edge, $\overline{V1V2}$, and start at verexV2 to find next vertex within $\Delta\phi$, (e) create concave edges, $\overline{CV2}$ and $\overline{CV3}$, because of not finding any photons, (f) continue to find next

vertex until we have finished searching all of photons.

Our extension includes several different steps from the *Graham's Scan* method. At the beginning, we must make sure that all of the photons in a group sit on the same flat surface. Because a group can have been merged together with other groups into a new group during the merging process, all of the photons in the new group could be distributed over different surfaces. It is hard and unnecessary to create a 3-dimentional polygonal boundary. Instead, we prefer to construct a 2-dimentional polygonal boundary by first projecting all photons onto the same plane as shown in Figure 4.7(a). We select a photon, C, which is nearest to the physical core position of the group, to be the pivot (this can be an internal point or a point on the boundary). The other photons then are projected onto the local uv-plane of C. Also, we sort these photons according to the size of the polar angle, ϕ, which is the angle between a photon and the u-axis.

Based on the assumption that the photons in a group correspond to *ray coherence*, we can expect that all of the N photons in a group are distributed uniformly. For a perfect photon distribution, any photon, except for photon C, should be able to find another photon within $\Delta\phi$, i.e., $2\pi/(N-1)$, as shown in Figure 4.7(c). If there is no photon found within $\Delta\phi$, it will result in a concave edge (see Figure 4.7(e)). In addition, to compensate for a possible error since the position of photon C diverges too much from the physical core position, we adopt a flexible parameter, m, which is a real number that ranges between 1.0 and 1.5, and multiply it by $\Delta\phi$, i.e., $2\pi m/(N-1)$. The parameter ensures that we can generate a larger $\Delta\phi$ and tend to create a convex rather than concave edge for the worst case. This case is often seen when photon C is on the boundary, $\Delta\phi$ could be insufficient to accurately construct a boundary.

4.5. Radiance Estimate

4.5.1. Advanced Nearest-Neighbor Density Estimation

In the previous photon pass, we have constructed a polygonal boundary for each photon group to represent the beam-like illumination area. Afterwards, we utilize the boundary information in the radiance estimate stage to enhance the strategy of sampling photons from the photon maps. Our method is first to detect which the polygonal boundaries of grouped photon maps intersect with query point x such as illustrated in Figure 4.8. We then locate the *N* nearest-neighbor photons around the query point from the photon kd-trees of these groups to estimate radiance.

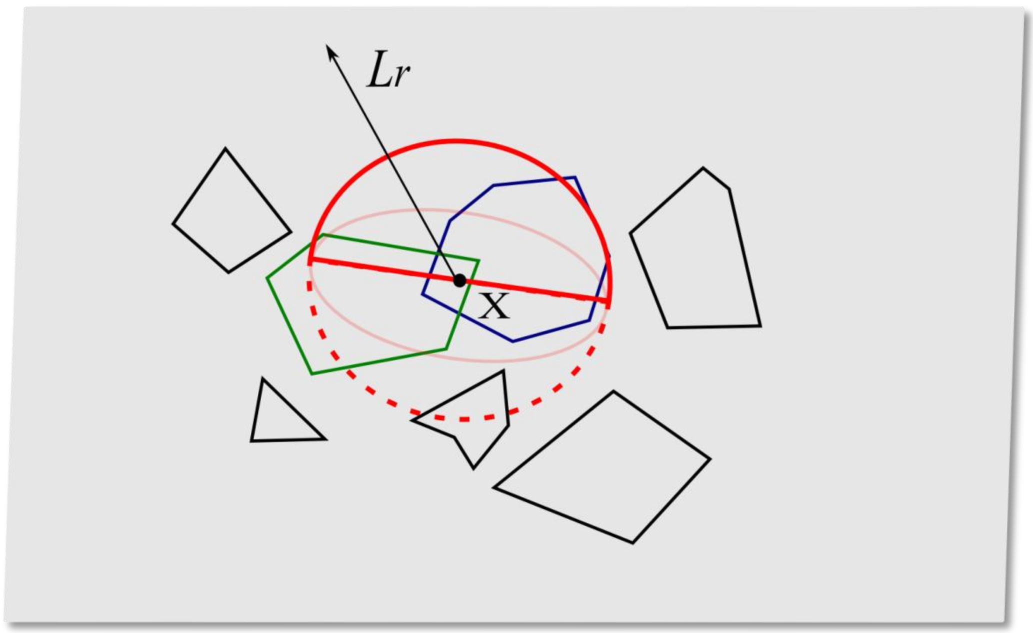

Figure 4.8. Performing the point-boundary intersection testing at query point x.

In implementation, the steps for performing point-boundary intersection testing are described as below:

a. In turn project the query point onto the surface where each group is deposited.

b. Compute the query point's relative polar angle ϕ to the local coordinate axis u of photon C (see Figure 4.7(a)).

c. Later find that the query point lies between which both vertices on the boundary by comparing the ϕ value such as illustrated in Figure 4.9 where we assume that query point X is between V_i and V_j.

d. If there is an edge between V_i and V_j, we need to further detect whether the point is inside the triangle consisting of V_i, V_j, and C by means of evaluating the dot product of vector V_x and N (a perpendicular vector to the edge (V_i, V_j)). (see Figure 4.9(a))

e. Otherwise, if there is no edge between V_i and V_j, the point must be not inside the boundary.(see Figure 4.9(b))

f. Until all of groups have been evaluated, we continue to perform the point-boundary intersection testing.

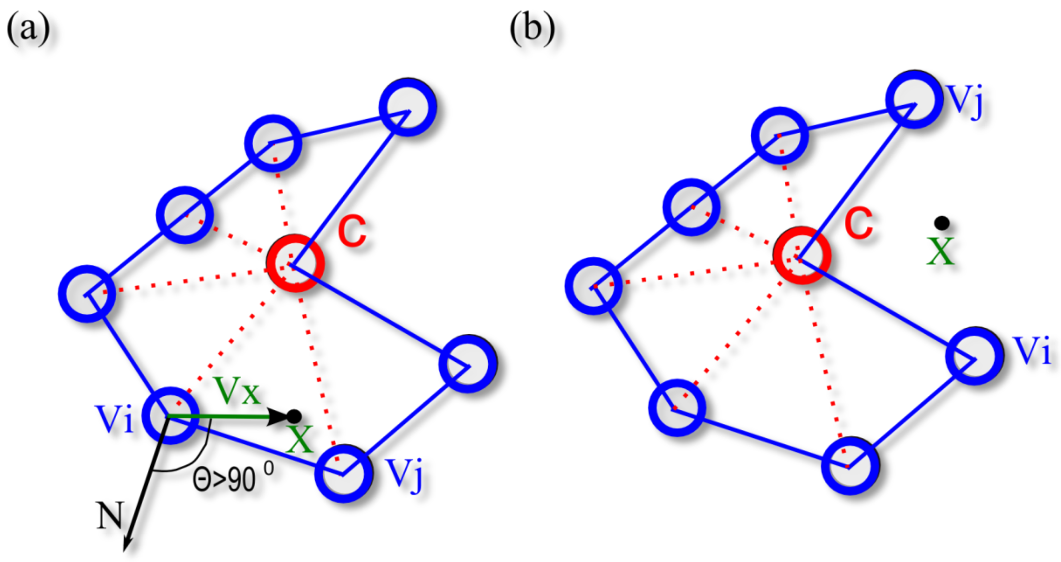

Figure 4.9. The point-boundary intersection testing: (a) there is an edge between V_i and V_j, where query point X hits the boundary because the dot (V_x, N) is less than zero ($\theta > 90°$) , (b) there is no edge such that X is not within the boundary.

Once we have found a group hit by the query point in the point-boundary intersection testing, we start to search for the *N* nearest-neighbor photons from its kd-tree. In addition, if there are at least two groups to be hit as shown in Figure 4.10, we in turn locate the nearest-neighbor photons from each group, until *N* closest or all photons (if the number of photons is smaller than the *N*) have been located.

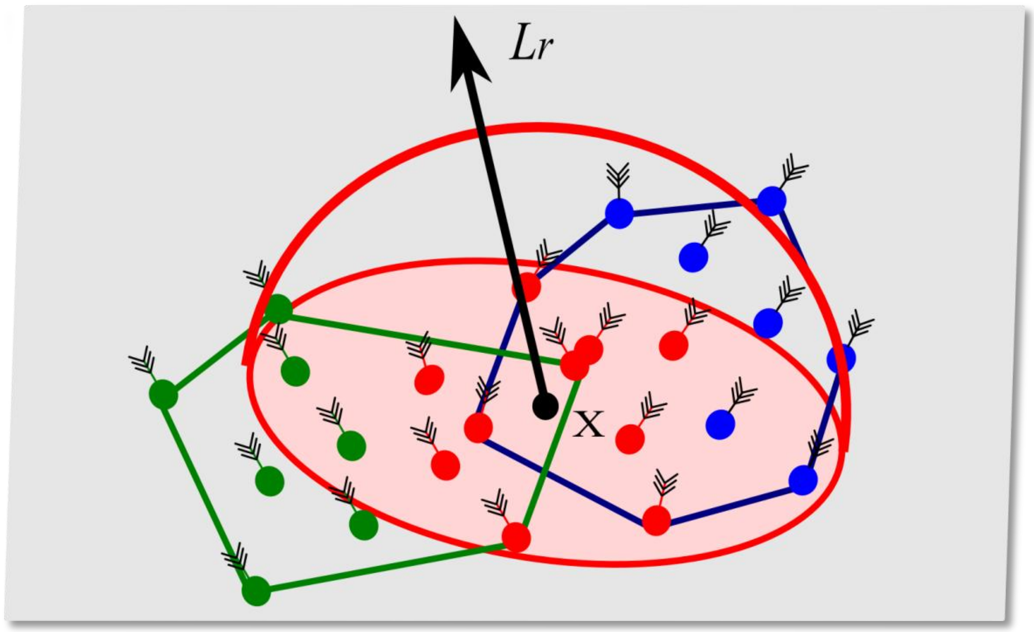

Figure 4.10. Locate 10 closest photons (red color) to the query point from two different photon groups and estimate the reflected radiance L_r.

In the radiance estimate stage, we adopt a smoother filter (k(p)), *Silverman kernel function* [14], to weight the energy of incident photons around the query point:

$$k(p) = 3\left(1 - \left(\frac{d_p}{d_{max}}\right)^2\right)^2 \Delta A^{-1} \approx 3\left(1 - \left(\frac{d_p}{d_{max}}\right)^2\right)^2 \Big/ \pi {d_{max}}^2 \tag{4.1}$$

Where p means one of the *N* closest photons to the query point. d_p represents the distance between p and the query point, and d_{max} is the maximum d_p. ΔA is the searching area which is denoted by the maximum disk area $\pi {d_{max}}^2$.

4.5.2. Double-Layer Kd-tree

In our density estimation method, the query point must first waste much time to perform a large number of point-boundary intersection tests so as to access the photon groups. However, in most cases, the query point only intersects with a limited number of groups. We, therefore, propose a double-layer kd-tree structure to efficiently exclude most of the groups that do not intersect with the query point certainly before performing the point-boundary intersection testing. (see Figure 4.11) For each photon group, we build a bounding sphere that can enclose all of the photons inside it. Afterwards, we use all of the center points of the bounding spheres to build a kd-tree, called the first-layer kd-tree. Meanwhile, under each tree node, we also maintain a second-layer kd-tree, which consists of the photons within a group. Since both kinds of kd-trees have a hierarchical relationship, we call them double-layer kd-tree.

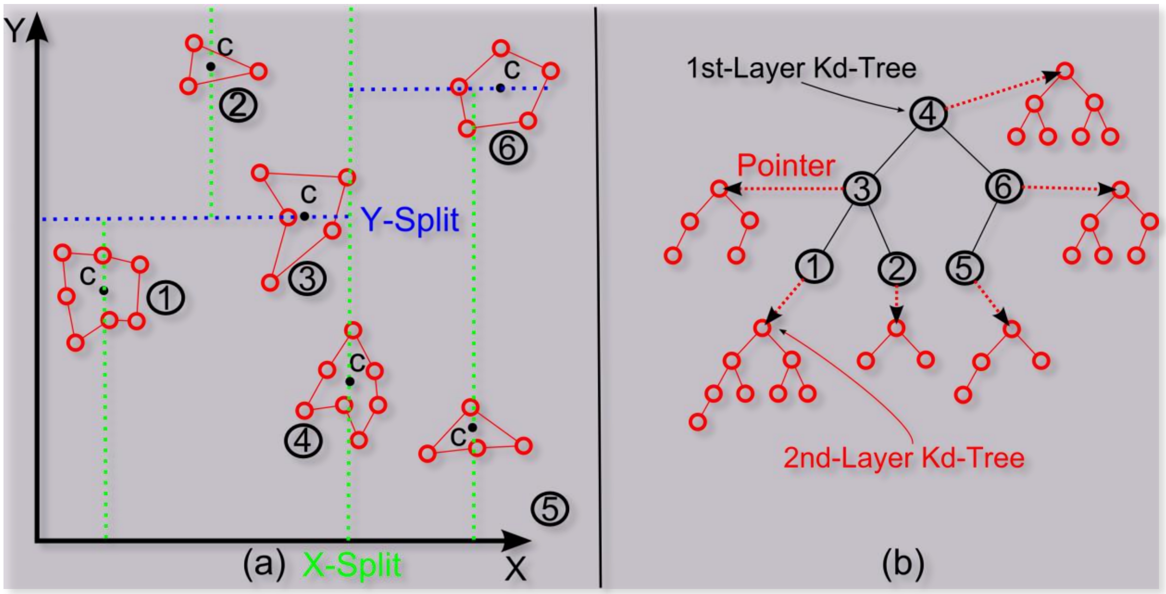

Figure 4.11. The double-layer kd-tree: (a) construct a first-layer kd-tree based on the center points of various groups, (b) Each node of the first-layer tree connects with a photon kd-tree from its group.

Based on the double-layer kd-tree structure, the sampling strategy has changed. We first define the searchable radius of the query point in the first-layer kd-tree as the radius of the maximum bounding-sphere. We then start to locate the first-layer tree nodes whose distances to the query point are smaller than the searching radius. If there is any node within the radius, we further evaluate whether that node's bounding sphere encloses the query point. Once the query point is inside this bounding sphere, it is necessary to perform an actual point-boundary intersection testing, as described in Section 4.5.1, to determine whether the query point can access this group's second-layer photon kd-tree.

Although the worst case is to perform all of the above-mentioned intersection testing, most of the photon groups have been quickly excluded during both simple point-sphere intersection tests. In addition, in comparison to the standard *photon mapping* method which stores all photons in a singular and large kd-tree, we only need to maintain a small-sized photon kd-tree under each first-layer tree node. This is helpful to efficiently search for the nearest-neighbor photons such that we can compensate for the excessive time spent on the point-boundary intersection testing.

4.6. Visualizing Global Illumination using Grouped Photon Maps

In order to visualize global illumination, we divide the *rendering equation*, as written in Equation 3.7, into four terms like Jensen's *photon mapping* (see Section 3.1.7~3.1.11): direct illumination, specualr or highly glossy reflection, caustic illumination, and soft indirect illumination. In this dissertation, we adopt different solutions to render direct and caustic illumination as mentioned in the following two paragraphs.

Although *Photon mapping* is efficient to render direct illumination by using direct and shadow photons to evaluate the visibility to light sources, it needs extra resources to store direct and shadow photons. Also, a large number of photons are needed to render a high-quality result in a complex scene. We, therefore, simply spawn shadow rays to test the visibility using Monte Carlo *importance sampling* based on both light sources and *BRDF* [41].

In comparison with soft indirect illumination, caustic illumination has a high variance such that it needs a large number of photons to render caustics with sharp or hard edge. So we adopt grouped photon maps to directly visualize caustic illumination instead of the photon maps. As mentioned in Section 4.4, we first use polygonal boundary to detect which groups are struck by the query point. The photon kd-trees of those struck groups are then accessed to search for the *N* nearest-neighbor photons and estimate the final caustic contribution.

As for soft indirect illumination, it is to evaluate the small contribution from indirect lighting which is diffusely reflected or transmitted at least once. As a result, it is enough to estimating radiance by using standard photon maps as mentioned in Section 3.1.11.

5. Results and Discussion

In this chapter, we have performed several experiments to test the caustics-based scenes and compare *grouped photon mapping* with *photon mapping* and *path tracing*. This chapter is divided into five sections: system and environment configuration, *level of detail* test, light leakage test, diamond caustics tests, and run-time statistics.

5.1. System and Environment Configuration

We implemented our algorithm on the PBRT rendering system [41] which offer some ray tracing-based techniques to render global illumination. The experiment was run on a PC with Intel Dual-Core CPU 2.66 GHz and main-memory of 4 GB.

We separately synthesized two kinds of image resolutions, including 512 by 512 pixels shown in Section 5.2 and Section 5.3, and 685 by 490 pixels shown in Section 5.4. The basic background scene includes a box similar to a *Cornell box*, and a diffuse square-area light source. Each one of the floor, walls, and ceiling use the same diffuse material “matte” (a reflectivity of 50%) which is built in the PBRT system. In addition, we set the maximal trace depth for all eye rays to 10, while shooting 16 sampling rays through each pixel of the viewing plane to average the result. The number of nearest-neighbor photons sampled around the query point is at most 50 except for the light leakage scene using 500 photons.

5.2. Level of Detail Test

In order to test the capability of *quadric error metrics* managing the process of merging groups, we rendered two scenes with caustic effect, including a glass sphere consisting of 32,512 meshes and a glass bunny with 69,451 meshes respectively. In both scenes, there were a total of about 100,000 photons stored in the grouped photon maps.

From Figure 5.1 to Figure 5.3, they show a series of synthesized images at different stages of surface simplification. In Figure 5.1, we barely see any caustic effect on the floor, because 95% of groups are recognized as noise groups which contain fewer than three predefined photons. These noise groups would be ignored in the rendering pass such that we cannot render any caustic effect. However, as the model is progressively simplified, more and more caustic effect obviously appears as shown in Figure 5.2 and Figure 5.3.

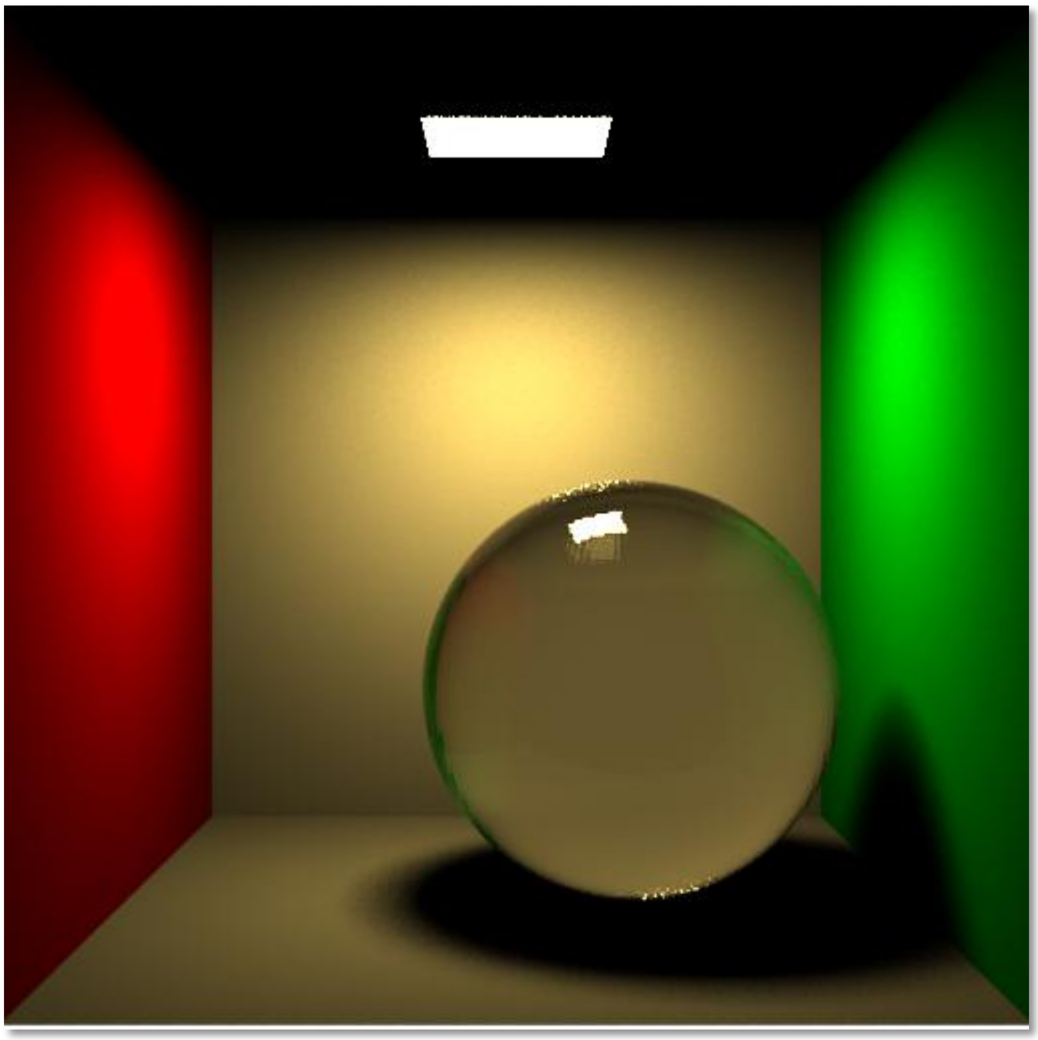

Figure 5.1. The caustic effect is rendered before performing any surface simplification.

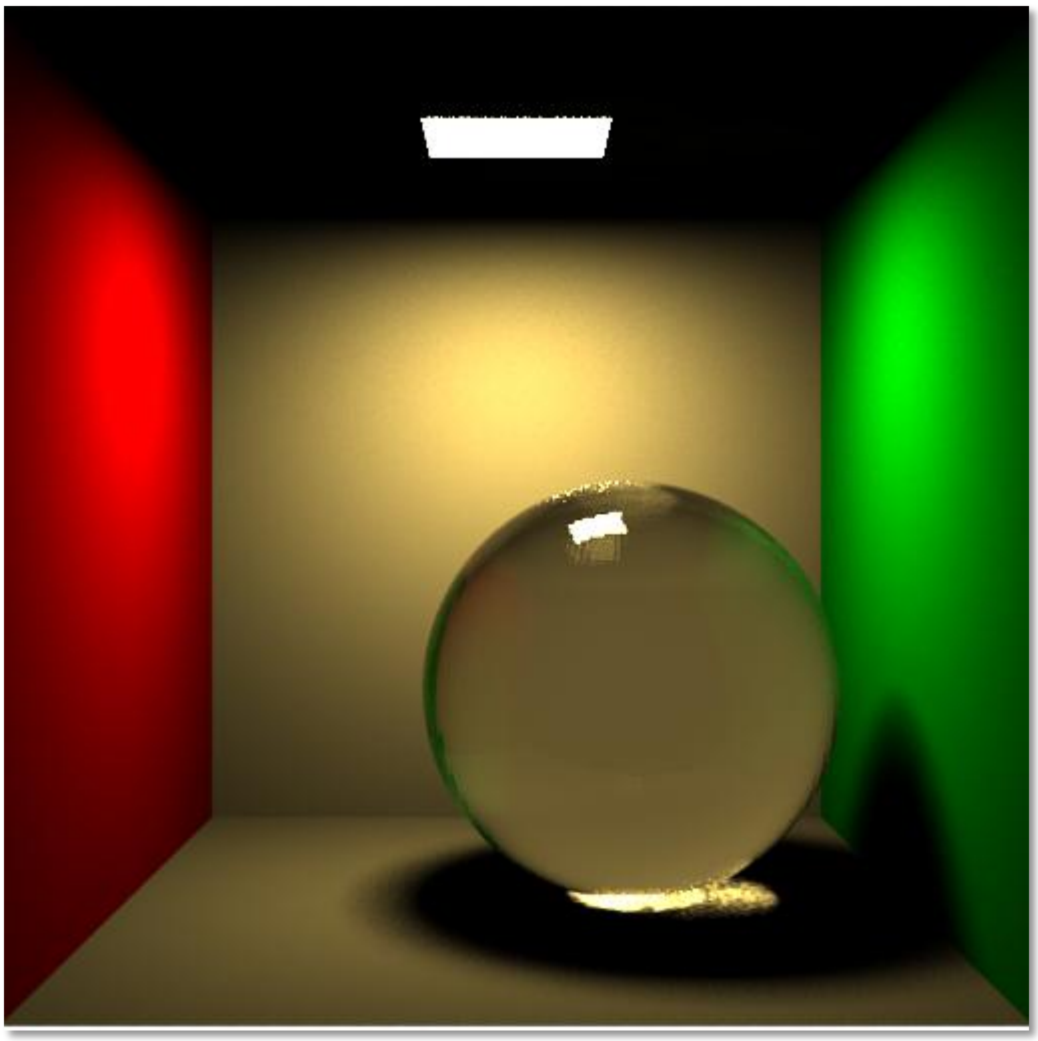

Figure 5.2. The caustic effect is rendered after reducing 15% of vertices.

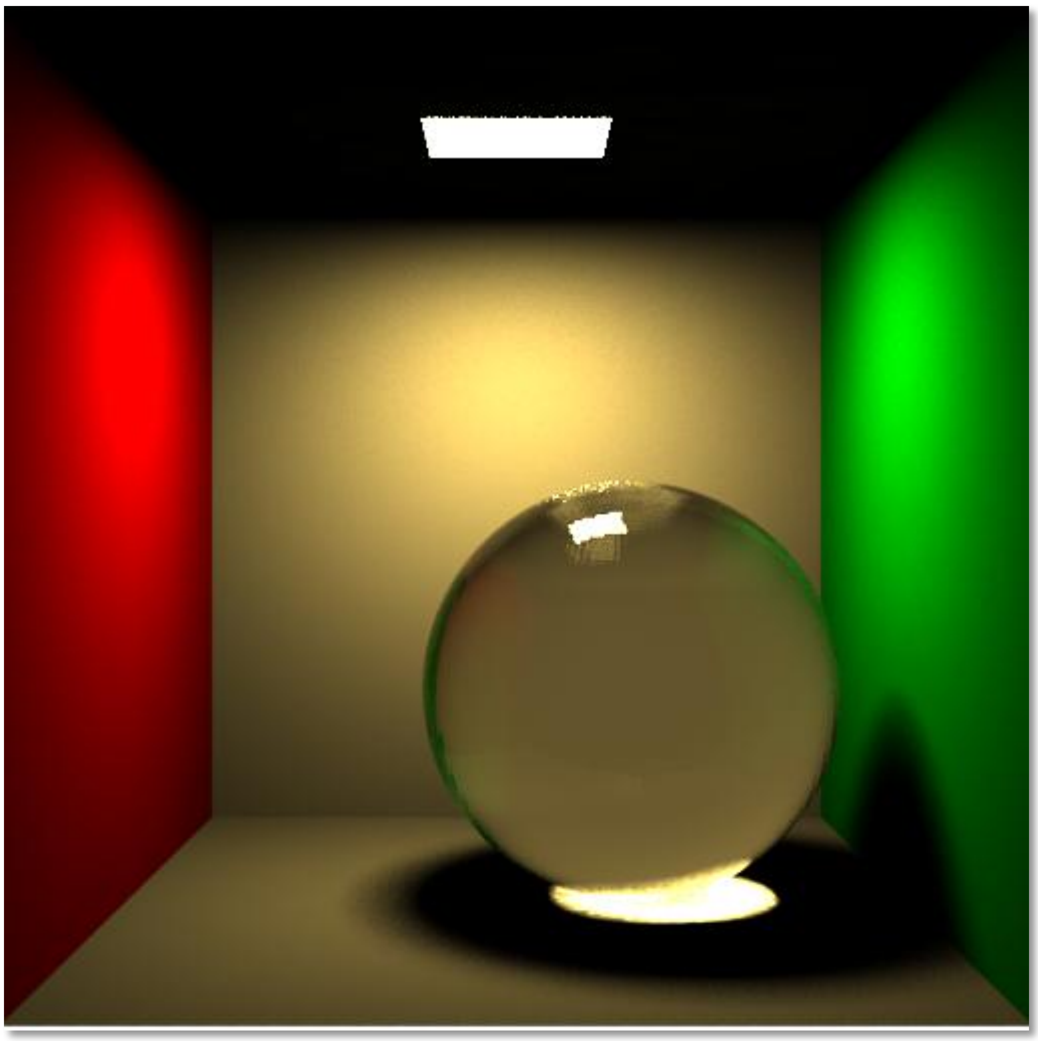

Figure 5.3. The caustic effect is rendered after reducing 30% of vertices.

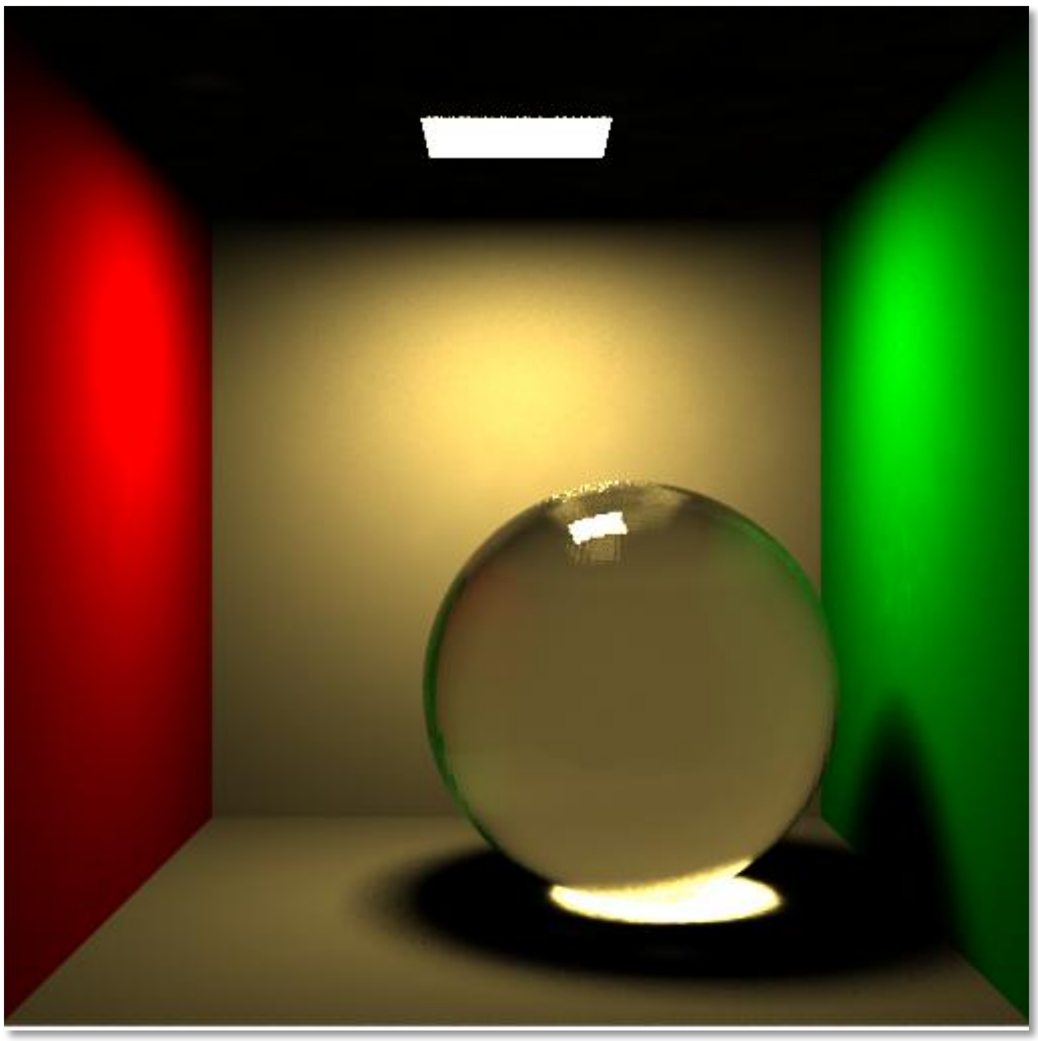

Figure 5.4. The caustic effect is rendered using standard *photon mapping*.

According to the ratio of valid-photons to total photons as illustrated in Figure 5.5 and Figure 5.6, which is the statistics of another glass bunny scene, we can also observe the above-mentioned growing tendency. In Figure 5.5, as vertex-pairs are collapsed or contracted at a rate of 5 percent each time, the valid-photon rate becomes higher and higher.

Also, it quickly converged into a steady status after about 30% of vertex-pairs to be contracted. That is to say, 90% of photons have been already merged into valid groups, and only 10% of invalid photons remain. In terms of efficiency, the contribution from these 10% of photons to the radiance is so small that we can ignore it and discontinue the surface simplification. This result can be verified by means of the image comparison between Figure 5.3 and Figure 5.4. We can clearly recognize that both images have almost the same quality of caustics, and even our method (Figure 5.3) generates clearer and sharper caustic edge. Therefore, this experiment proves that we can successfully achieve good image quality and reach a better trade-off between noise and bias, while moderately merging groups by the *quadric error metrics* technique.

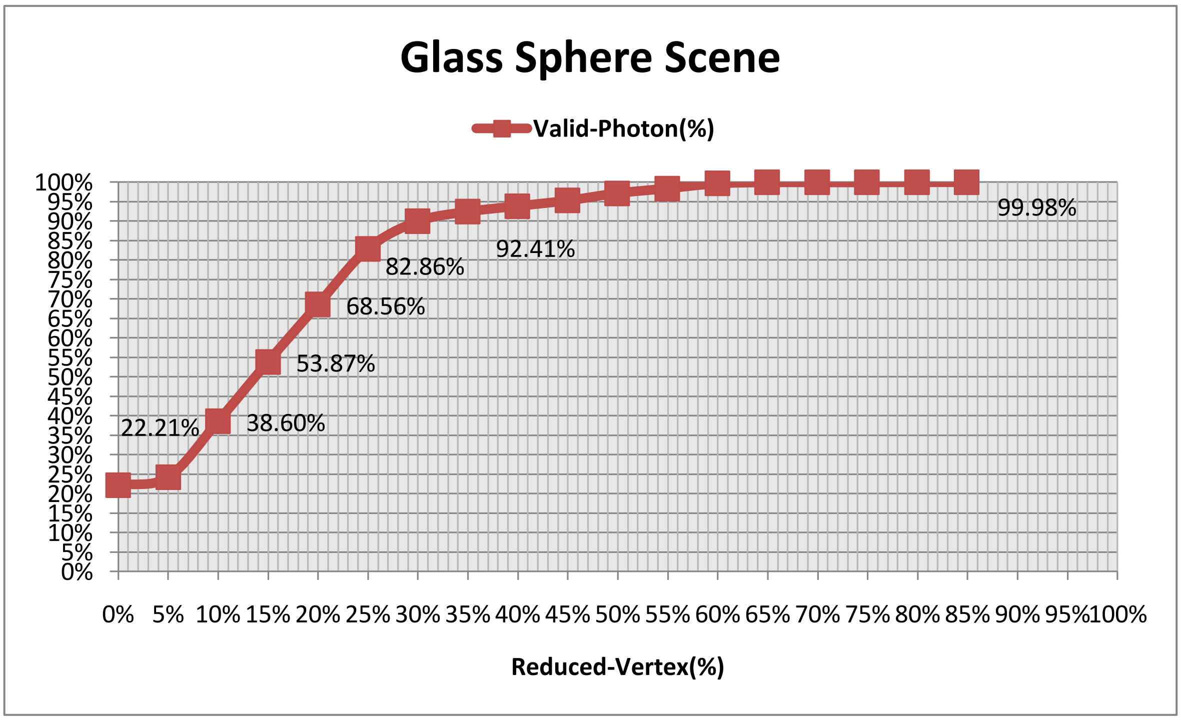

Figure 5.5. Valid-photon (%) vs. Reduced-Vertex (%) for the glass sphere scene. The curved line converges quickly after contracting about 30% of vertices.

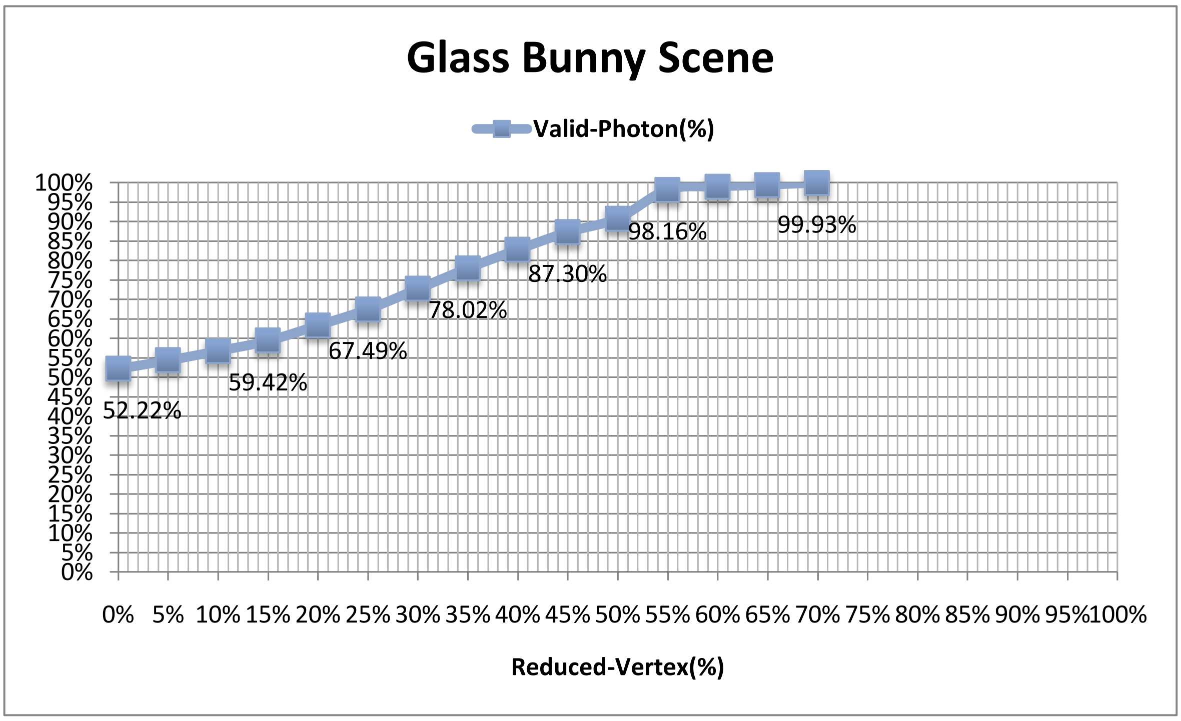

Figure 5.6. Valid-photon (%) vs. Reduced-Vertex (%) for the glass bunny scene. The curved line progressively grows and converges after contracting about 50% of vertices.

5.3. Light Leakage Test

In the third test scene as shown in Figure 5.7 and Figure 5.8, we demonstrate a box scene, which is divided into left and right rooms, to test the problem of light leakage. In this scene, the left wall of the left room consists of a mirror material surface. Also a square light source with a warm color is placed on the left room’s ceiling, while a point light source with a cool color is only added to light up the right room. In the photon pass, we shower about 100,000 photons from the warm light into the scene. In addition, in order to emphasize the light leakage effect, we make the query point possible to locate at most 500 closest photons.

Figure 5.7 uses the *photon mapping* method to render the scene. We find that the warm light leaks from the left room to the wall corners of the right room. This is because the query point, which approaches to the right side of the middle wall, has sampled the invisible photons from the left room. On the contrary, our *grouped photon mapping*

(Figure 5.8) completely eliminated light leakage due to the usage of polygonal boundaries to exclude invisible photons to the eyes.

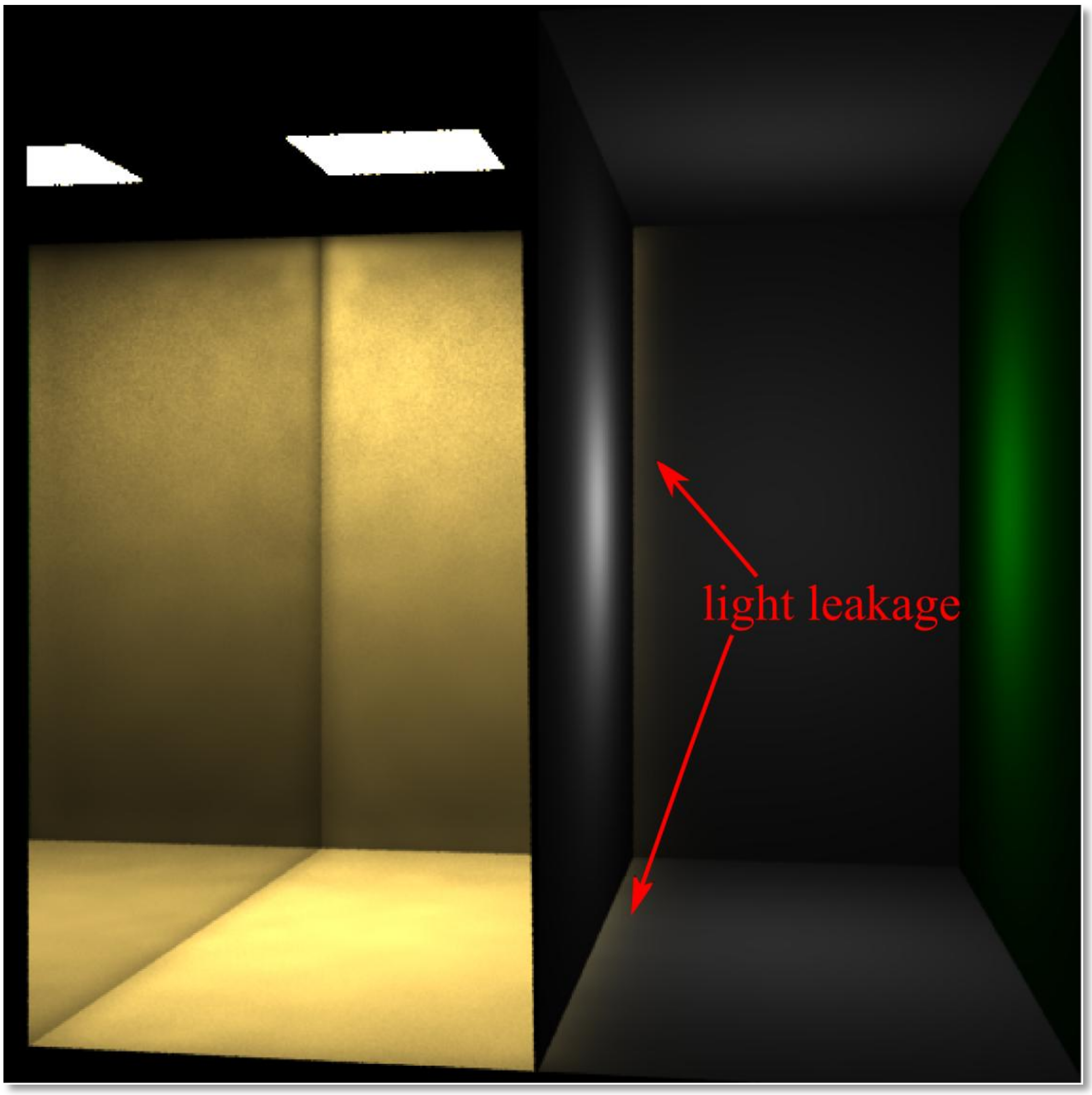

Figure 5.7. Light leaks from left room into right room while using *photon mapping*.

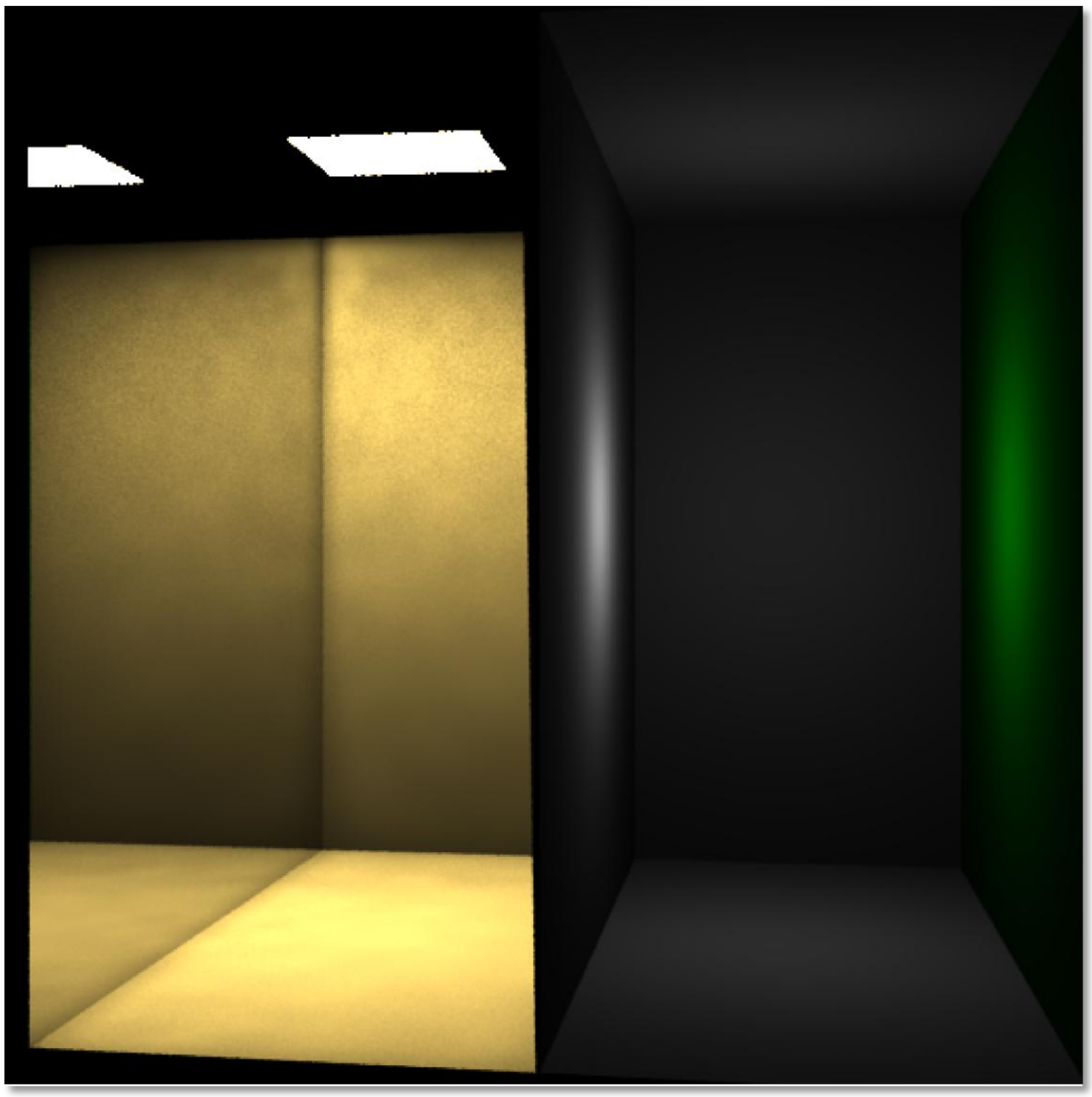

Figure 5.8. There is no light leakage while using *grouped photon mapping*.

5.4. Diamond Caustics Test

In this experiment, we test a series of diamond scenes with complex optical effect to compare our method with *photon mapping* and *path tracing*. Diamond, like many other gemstones, has a specular surface with a very high *refractive index*. This would intensely dispersed light into different monochromatic light while light goes through the diamond. In addition, for the diamond whose shape and appearance are perfectly cut and polished, this reason has light rays tend to be reflected and refracted multiple times inside the object. As a result, a great deal of bright and hard edge of caustics with fire effect is projected onto the floor.

At the beginning, we render a round brilliant cut of diamond scene while one million of caustic photons are stored in the grouped photon maps. The synthesized images from Figure 5.9 to Figure 5.11 are rendered by *photon mapping*, *grouped photon mapping*, and *path tracing* respectively.

We clearly observe that *photon mapping* generates a great deal of blurry edge and dot-shaped caustics. In comparison with a realistic photograph as shown in Figure 1.2, this result loses the original caustic shape and sharp edge. The problem results from the fact that *photon mapping* cannot filter out the photons outside the edge of caustics. That is to say, although these false photons are outside the edge, they are also inside the searching area of the query point such that they are mis-sampled. In contrast, our method demonstrates a clearer and sharper edge of caustics as shown in Figure 5.10. The reason comes from adopting a nature and adaptive searching area consisting of polygonal boundaries to exclude those false photons. We also compare with traditional *path tracing* algorithm, such as the image shown in Figure 5.11. It also yields an undesirable result, even after raising the number of rays to over 20,000 per pixel at the cost of 12 hours and 35 minutes totally. This is because the complex optics like light dispersion and multiple reflections or refractions makes *path tracing* hard to handle all of complex paths.

We also render another three different cutting shapes of diamond scenes, including emerald-shaped, pear-shaped and multiple diamonds. These images are shown in Figure 5.12, Figure 5.13, and Figure 5.14 respectively.

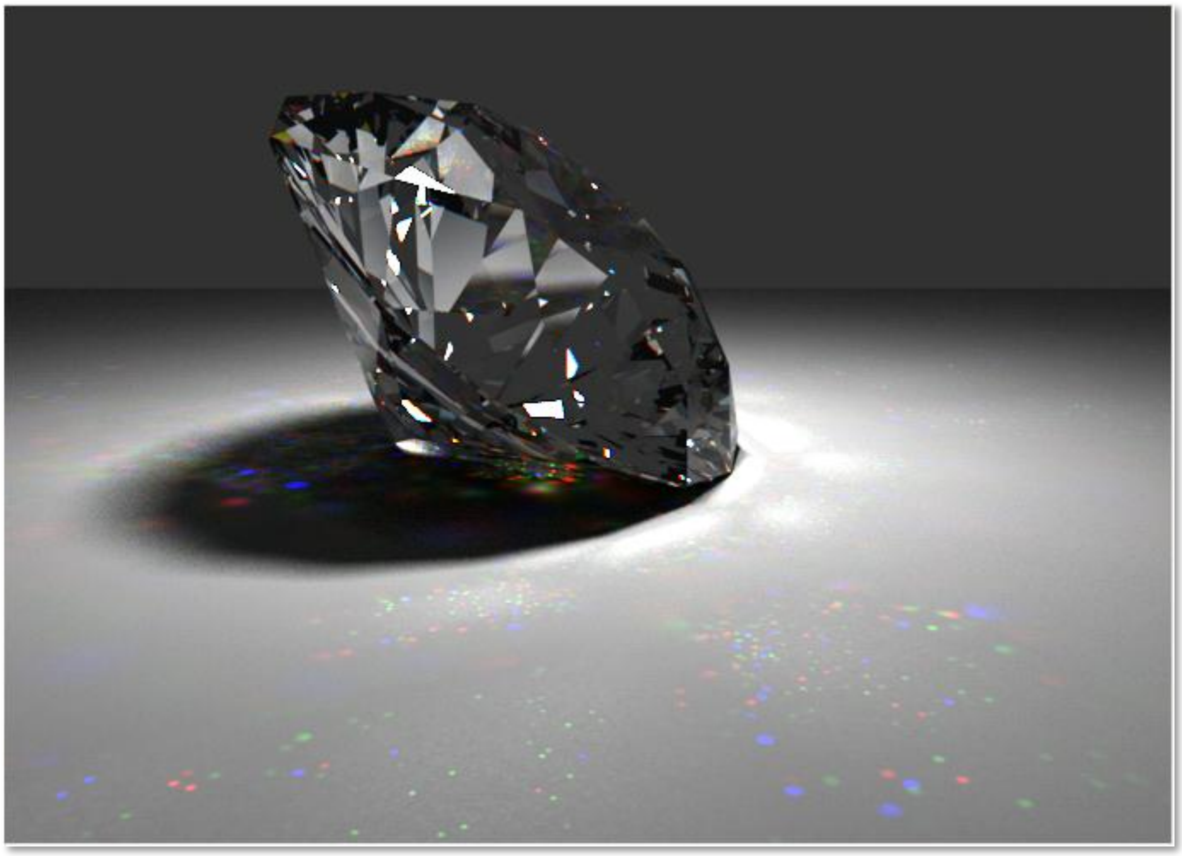

Figure 5.9. A round brilliant cut of diamond with complex caustics is rendered using *photon mapping*.

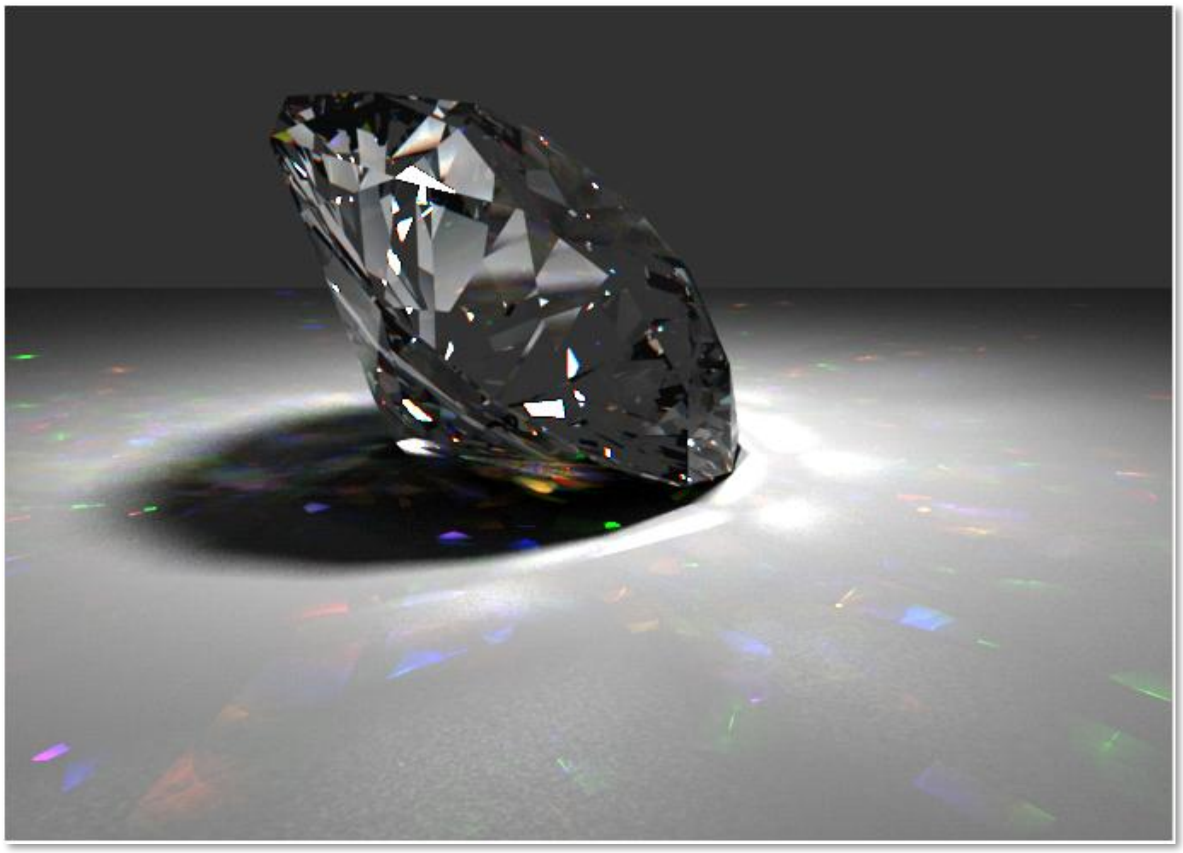

Figure 5.10. A round brilliant cut of diamond with complex caustics is rendered by *grouped photon mapping*.

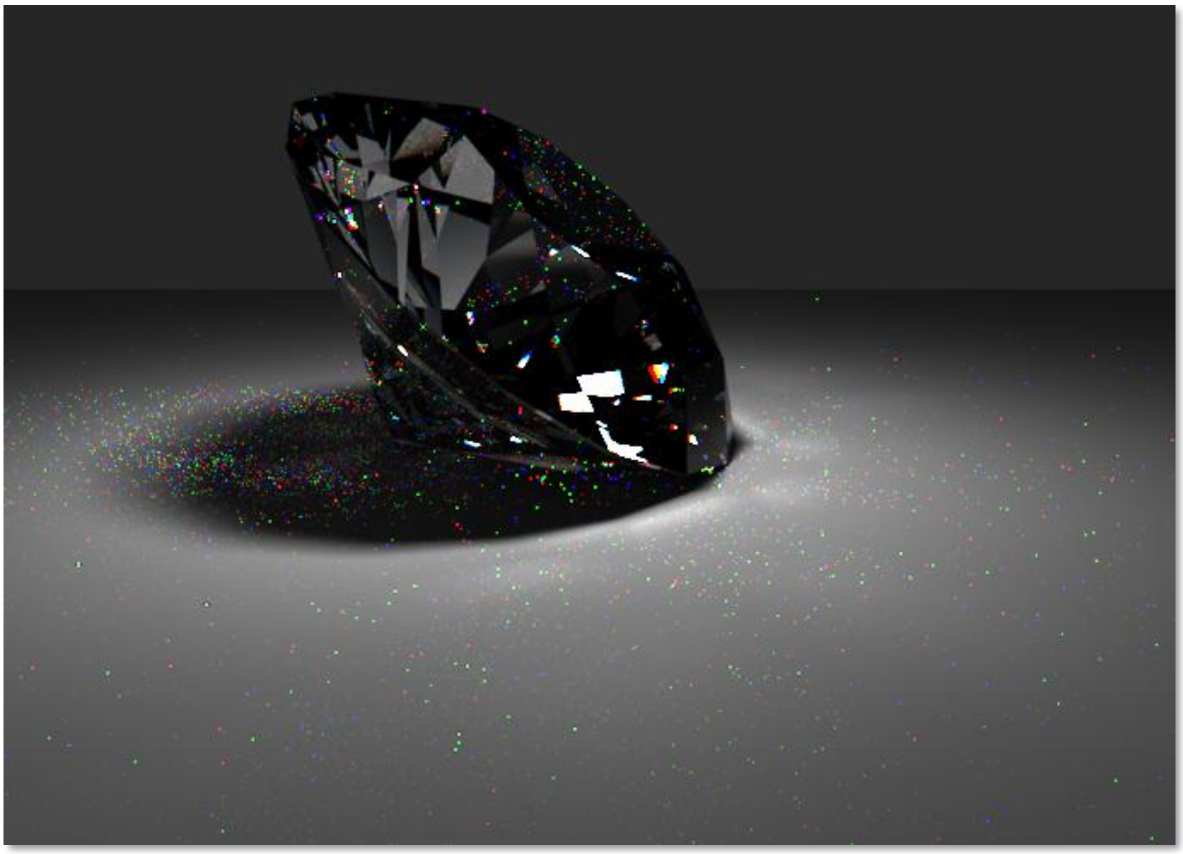

Figure 5.11. A round brilliant cut of diamond with complex caustics is rendered using *path tracing*.

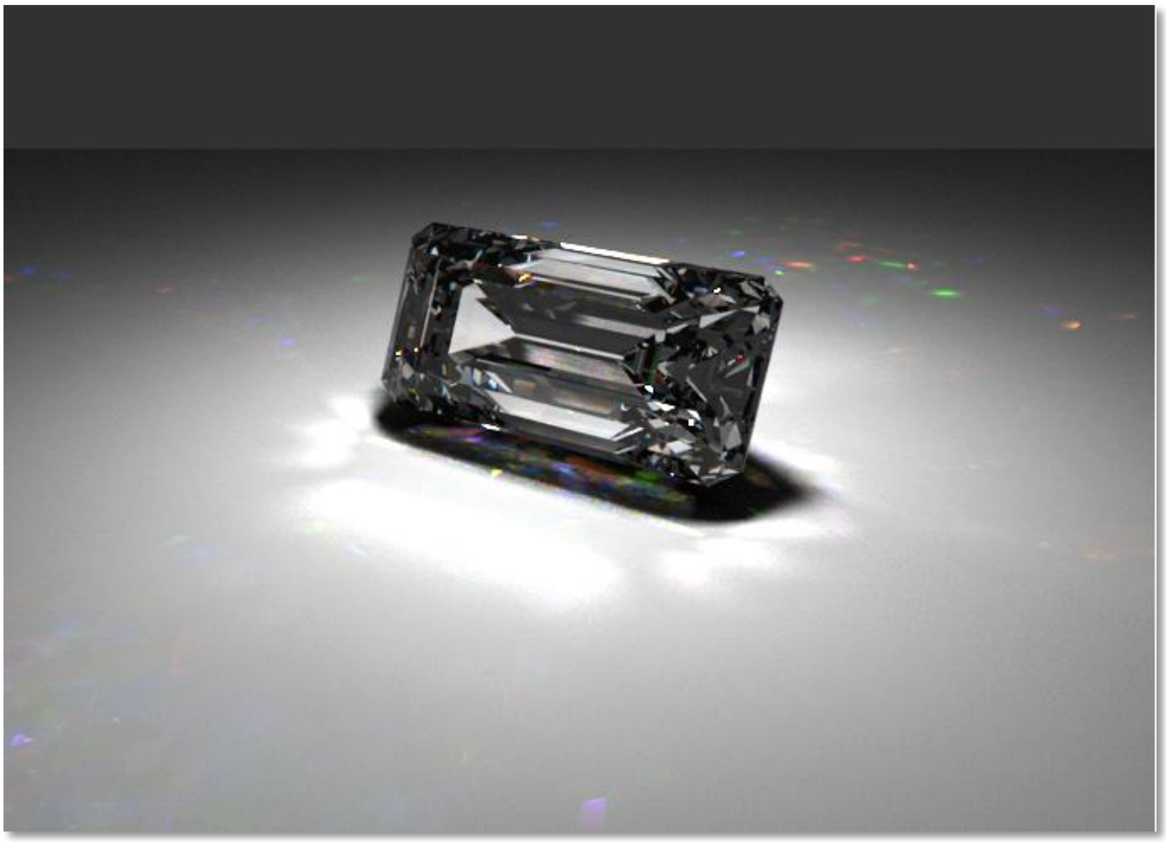

Figure 5.12. Emerald-cut diamond is rendered using *grouped photon mapping*.

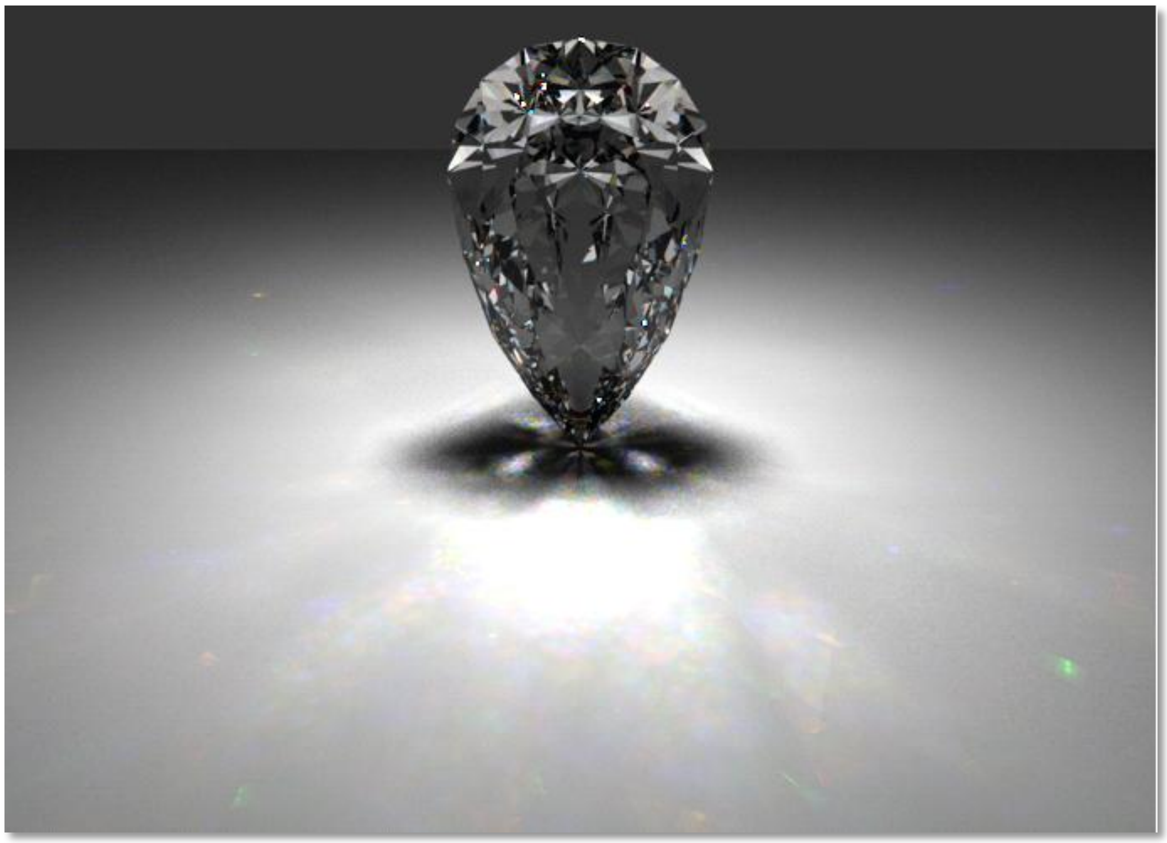

Figure 5.13. Pear-cut diamond is rendered using *grouped photon mapping*.

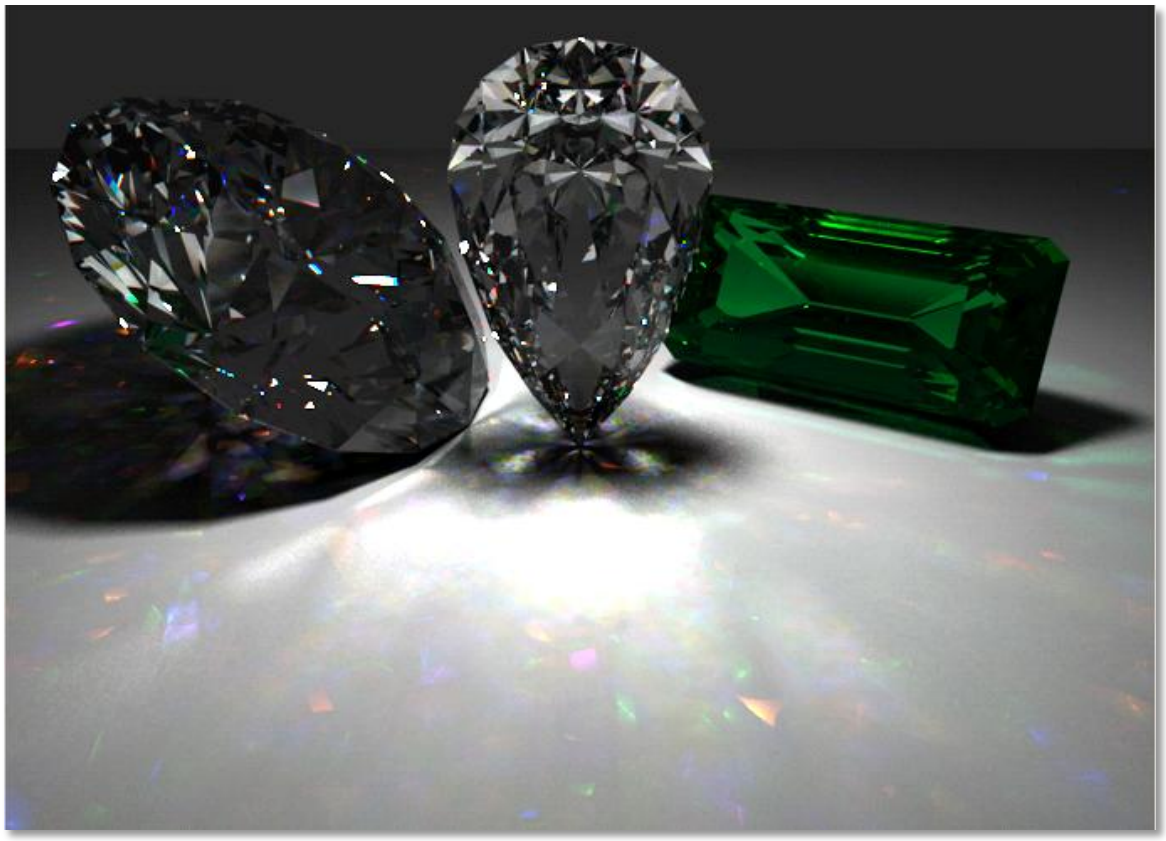

Figure 5.14. Multiple diamonds are rendered using *grouped photon mapping*.

5.5. Run-Time Statistics

Table 5.1 lists the run-time statistics of three types of scenes, including a glass sphere (Figure 5.3 and Figure 5.4), a two-room box (Figure 5.7 and Figure 5.8), and a diamond (Figure 5.9 and Figure 5.10). In the first case, both *photon mapping* and *grouped photon mapping* provide acceptable image results, though the latter takes more time for clustering, merging, and point-boundary intersection testing. In the second case, light leakage is eliminated successfully by *grouped photon mapping*. Moreover, the rendering time is significantly lesser than that of *photon mapping* because the bounding sphere excluded most of the groups and photons beforehand. In the last case, the accuracy and quality of the diamond image are obviously improved by *grouped photon mapping* at only a small additional time cost.

Table 5.1. Run-time statistics. In the GPM method, the merging time costs of two-room-box and diamond scenes are both zero because their initial percentages of valid-photons are already higher than the default threshold of 90%

Scene	**Method**	**Cluster Time (sec)**	**Merge Time (sec)**	**Photon Pass Time (sec)**	**Render Time (sec)**
Glass Sphere	PM	-	-	18	454
	GPM	138.07	302.77	460.6	919.2
Two-box Room	PM	-	-	2	272.1
	GPM	0.02	-	2.02	126.6
Diamond	PM	-	-	875.4	547.7
	GPM	0.4	-	880.6	592.4

6. Conclusion and Future Work

6.1. Conclusion

In this dissertation, we have proposed a novel global illumination approach that integrates the *photon mapping* method with the light-beam concept to improve the nearest-neighbor density estimation method.

Initially we cluster all of the photons into different groups based on the intersection-history coherence so as to rebuild the beam-like illumination area from each photon group. Also, we utilize *quadric error metrics* algorithm to progressively manage the process of merging noise groups and try to obtain a trade-off between noise and bias.

In the rendering pass, the above-mentioned illumination area serves as a filter to restrict the searchable area of a query point for locating the *N* nearest-neighbor photons. Furthermore, we combine this bounding area information with a double-layer kd-tree structure to efficiently reduce the time for performing the point-boundary intersection testing and searching for the closest photons to the query point.

The experiment results show that our algorithm successfully renders higher quality results than the standard *photon mapping* method. Under *quadric error metrics*, some visible noise is appropriately eliminated and the sharp or hard edge of illumination features such as caustics is accurately generated to reduce *proximity bias*.

By the way of maintaining the polygonal boundaries for different beam-like groups, the searchable area around the query point can be effectively used to exclude the invisible photons to the eyes. This can achieve the purpose of eliminating light leakage.

Finally, our advanced nearest-neighbor density estimation method also successfully renders a diamond scene with complex optical effects. The hard edge of caustics can be not only displayed clearly, but the original caustic shape can be also maintained smoothly. Both effects cannot be accurately rendered using the *photon mapping* method.

6.2. Future Work

Although our algorithm effectively enhances the image quality, several problems remain to be solved.

Most of the time used to build the grouped photon maps is wasted on the processes of clustering photons and merging groups. More research effort is necessary to find a better way or data structure to record the *intersection history* so that we can quickly organize the photons.

In our algorithm, we adopt a continuous *level of detail* technique to manage the process of merging groups. However, when the scene needs a large amount of surface simplification, this will bring a heavy load which needs wasting time on surface simplification and performing the time-consuming merging process multiple times. In the future, we can first prepare different *level-of-detail* versions in the preprocess stage and find a relationship between the number of photons and object models to efficiently lower the frequency of the merging process.

When the number of showered photons is insufficient, *grouped photon mapping* tends to create a visible defect. As shown in the top of Figure 6.1, a dark area is incorrectly rendered around the corner, when there are only 10,000 photons used. This problem occurs because *grouped photon mapping* fails to construct enough polygonal boundaries to lean up against the edge and corners. Relatively, raising the number of photons to 20,000 can effectively alleviate this problem. For future work, we are considering to adaptively adjust the number of photons by introducing a preprocessing stage which calculates the photon density at the edge of a scene.

Currently, *grouped photon mapping* is only applied to the scenes that mainly consist of polygonal models. In the future, we hope to extend our method to other types of models by exploiting a more general-purpose architecture.

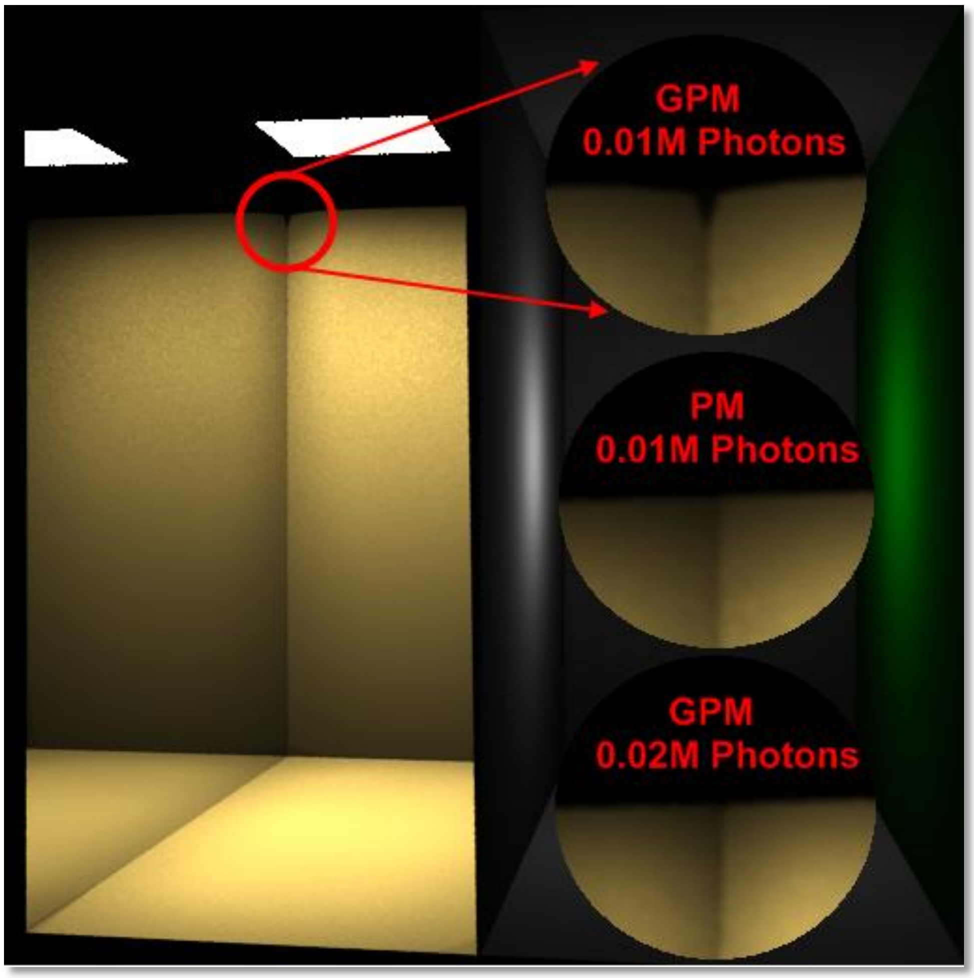

Figure 6.1. A visible dark edge appears around the corner while *grouped photon mapping* only showers very small number of photons (0.01 million photons) into the scene.

Bibliography

[1] Ward, G. J., Heckbert, P., “Irradiance gradients”, in *Proceedings of the Third Eurographics Workshop on Rendering*, pp. 85-98, 1992.

[2] Veach, E., Guibas, L. J., “Bidirectional estimators for light transport”, in *Proceedings of the Fifth Eurographics Workshop on Rendering*, pp. 147-162, 1994.

[3] Lafortune, E., Willems, Y., “A 5D tree to reduce the variance of Monte Carlo ray tracing”, in *Rendering Techniques '95*, pp. 11-20, 1995.

[4] Veach, E. and Guibas, L. J., “Metropolis light transport”, in *Proceedings of the 24th Annual Conference on Computer Graphics and interactive Techniques*, pp. 65-76, 1997.

[5] Smits, B., Arvo, J., and Greenberg D., “A clustering algorithm for radiosity in complex environments”, in *Proceedings of the 21st Annual Conference on Computer Graphics and interactive Techniques SIGGRAPH '94*, pp. 435-442, 1994.

[6] Christensen, P. H., Stollnitz, E. J., Salesin, D. H., and DeRose, T. D., “Global illumination of glossy environments using wavelets and importance”, in *ACM Transaction on Graphics*, vol. 15, no. 1, pp. 37-71, 1996.

[7] Arvo, J., “Backward ray tracing”, in *Developments in Ray Tracing*, SIGGRAPH Course Notes, vol. 12, pp. 259-263, 1986.

[8] Chen, S. E., Rushmeier, H. E., Miller, G., and Turner, D., “A progressive multi-pass method for global illumination”, in *Proceedings of the 18th Annual Conference on Computer Graphics and interactive Techniques SIGGRAPH '91*, pp. 165-174, 1991.

[9] Zimmerman, K., and Shirley, P., “A two-pass solution to the rendering equation with a source visibility preprocess”, in *Rendering Techniques '95*, pp. 284-295, 1995.

[10] Jensen, H. W., “Global illumination using photon maps”, in *Rendering Techniques '96*, pp. 21-30, 1996.

[11] Schregle, R., “Bias compensation for photon maps”, in *Computer Graphics Forum*, vol. 22, no. 4, pp. 729-742, 2003.

[12] Jensen, H. W., “The photon map in global illumination”, Ph.D. dissertation, Technical University of Denmark, 1996.

[13] Walter, B., “Density estimation techniques for global illumination”, *Technical Report*, Cornell University, TR98-1700, 1998.

[14] Shirley, P., Wade B., Hubbard P., Zareski D., Walter B. and Greenberg D., “Global illumination via density estimation”, in *Proceedings of the 6th Eurographics Workshop on Rendering*, pp. 187-199, 1995.

[15] Jensen, H. W. and Christensen, N. J., “Photon maps in bidirectional Monte Carlo ray tracing of complex objects”, in *Computers & Graphics*, Elsevier, vol. 19, no. 2, pp. 215-224,1995.

[16] Jensen, H. W., “Realistic image synthesis using photon mapping”, A. K. Peters, 2001.

[17] Hey, H., and Purgathofer, W., “Global illumination with photon mapping compensation”, *Technical Report*, Vienna University of Technology, TR-186-2-01-04, 2001.

[18] Tobler, R.F., and Maierhofer, S., “Improved illumination estimation for photon maps in architectural scenes”, in *Proceedings of WSCG 2006*, pp. 257-262, 2006.

[19] Lastra, M., Urena, C., Revelles, J., and Monyes, R., “A particle-path based method for Monte Carlo density estimation”, in *Proceedings of the 13th Eurographics Workshop on Rendering*, pp. 33-40, 2002.

[20] Havran, V., Bittner, J., Herzog, R., and Seidel, H. P., “Ray maps for global illumination”, in *Proceedings of Eurographics Symposium on Rendering 2005*, pp. 43-54, 2005.

[21] Herzog, R., Havran, V., Kinuwaki, S., Myszkowski, K., and Seidel, H. P., “Global

illumination using photon ray splatting", in *Computer Graphics Forum*, vol. 26, no. 3, 503-513, 2007.

[22] Hachisuka, T., Ogaki, S., and Jensen, H. W., "Progressive photon mapping", in *ACM SIGGRAPH Asia 2008 Papers*, 2008.

[23] Hachisuka, T. and Jensen, H. W., "Stochastic progressive photon mapping", in *ACM SIGGRAPH Asia 2009 Papers*, 2009.

[24] Garland, M., and Heckbert, P. S., "Surface simplification using quadric error metrics", in *Proceedings of SIGGRAPH 97*, pp. 209-216, 1997.

[25] Bentley, J. L., "Multidimensional binary search trees used for associative searching", in *Communication of ACM*, vol.18, no. 9, 1975.

[26] Bentley, J. L., "Multidimensional binary search trees in database applications", in *IEEE Transactions on Software Engineering*, vol. 5, no. 4,p.p. 333-340, 1979.

[27] Bentley, J. L. and Friedman, J. H., "Data structures for range searching", *ACM Computing Survey*, vol. 11, no. 4, p.p. 397-409, 1979.

[28] Silverman, B. W., "Density Estimation", Chapman and Hall, London, 1986.

[29] Kajiya, J. T., "The rendering equation", in *Proceedings of the 13th Annual Conference on Computer Graphics and interactive Techniques*, SIGGRAPH '86, p.p. 143-150, 1986.

[30] Arvo, J. and Kirk, D., "Particle transport and image synthesis", in *Proceedings of the 17th Annual Conference on Computer Graphics and interactive Techniques*, SIGGRAPH '90, p.p. 63-66,1990.

[31] Ward, G., "Real Pixels", in *Graphic Gems II*, Academic Press Professional, pp.80-83, 1991.

[32] Sung, K. and Shirley, P., "Ray tracing with the BSP tree", in *Graphics Gems III*, Academic Press Professional, p.p. 271-274, 1992.

[33] Nicodemus, F. E., Richmond, J. C., Hsia, J. J., Ginsberg, I. W., and Limperis, T., "Geometrical considerations and nomenclature for reflectance", in *Physics-Based Vision: Principles and Practice: Radiometry*, Jones and Bartlett Publishers, p.p. 94-145, 1992.

[34] Cook, R. L., Porter, T., and Carpenter, L., "Distributed ray tracing", in *ACM SIGGRAPH Computing Graphics*, vol. 18, no. 3, p.p. 137-145, 1984.

[35] Jensen, H. W., "Importance driven path tracing using the photon map", in *Rendering Techniques '95*, Springer Verlag, p.p. 326-335, 1995.

[36] Clark, J. H., "Hierarchical geometric models for visible surface algorithms", in *Communication of ACM*, vol. 19, no. 10, p.p. 547-554, 1976.

[37] Luebke D., Reddy M., Cohen J., Varshney A., Watson B., and Huebner R., "Level of Detail for 3D Graphics", Morgan Kaufmann, 2002.

[38] Garland M., "Quadric-Based polygonal surface simplification", Ph.D. dissertation, Carnegie Mellon University, 1999.

[39] Heckbert P. S., "Adaptive radiosity textures for bidirectional ray tracing", in *Computer Graphics*, vol. 24, no. 4, p.p. 145-154, 1990.

[40] Graham, R.L., "An efficient algorithm for determining the convex hull of a finite planar set", in *Information Processing Letters*, vol. 1, no. 4, p.p. 132-133, 1972.

[41] Pharr, M., and Humphreys, G., "Physically based rendering from theory to implementation", Morgan Kaufmann, 2004.