Universidade Nova de Lisboa

Faculdade de Ciências e Tecnologia

Departamento de Informática

Modelo de Iluminação Global por Radiação (Radiosidade)

Pedro Filipe Quintas Gomes

Dissertação apresentada na Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa, para a obtenção do grau de Mestre em Engenharia Informática.

Orientador: Professor Doutor Manuel João Toscano Próspero dos Santos

Fevereiro de 2010

N.º do aluno: 26889

Nome: Pedro Filipe Quintas Gomes

Título da dissertação:
Modelo de Iluminação Global por Radiação (Radiosidade)

Palavras-Chave:

- Iluminação Global
- Radiação Térmica
- Algoritmo Computacional
- Factor Forma
- Computação Gráfica

Keywords:

- Global Illumination
- Thermal Radiation
- Computer Algorithm
- View Factor
- Computer Graphics

Agradecimentos

Gostaria de começar por agradecer à minha família, que ao longo desta dissertação sempre acreditaram em mim, mesmo nos momentos mais difíceis, onde o sentimento de dúvida se instalava. Sem o seu apoio, todo este esforço teria sido em vão.

Queria agradecer especialmente ao meu orientador, o Professor Próspero dos Santos. Para além do seu profissionalismo inquestionável, é uma pessoa afável e simpática que me ajudou muitíssimo durante esta dissertação. Foi ele que me deu a chave para abrir a porta da área da Computação Gráfica, cabe-me a mim abri-la e usufruir da imensidão de conhecimento que está por detrás dela.

Não poderia terminar sem referir os meus colegas e amigos. A vossa ajuda foi preciosa, tornou esta experiência ainda mais agradável.

No momento de agradecer a todos aqueles que ajudaram como puderam e às vezes sem poderem, estas simples palavras não vos fazem justiça.

Resumo

Os modelos de iluminação global deram um contributo muito substancial na obtenção de ambientes virtuais foto-realistas em Síntese de Imagem, particularmente o Modelo de Iluminação Global por Radiosidade. Este modelo de iluminação marcou um ponto de viragem na área da Síntese de Imagem, pois recorre ao modelo de transferência de energia entre superfícies proveniente da área da Termodinâmica. Como a luz é uma forma de energia, este modelo pode ser aplicado em Síntese de Imagem para caracterizar, de forma realista, o comportamento da luz.

Em tempos, os elevados recursos computacionais necessários para aplicar este modelo eram um factor dissuasor. Hoje em dia, os computadores pessoais dispõem de hardware gráfico suficientemente genérico e com uma capacidade computacional que possibilita a utilização deste modelo.

Esta dissertação pretende analisar o estado da arte do Modelo de Iluminação Global por Radiosidade, concebendo e implementando um protótipo capaz de aplicar este modelo de iluminação usando hardware gráfico comum.

Para poder aplicar este modelo é necessário preparar o ambiente virtual. Esta preparação tem um grande impacto na qualidade dos resultados obtidos. Consiste normalmente em subdividir a geometria do ambiente virtual. Esta subdivisão dita os potenciais erros de aproximação. Mas é inevitável, uma vez que é impossível aplicar o modelo na sua forma analítica. O protótipo irá permitir ao utilizador especificar o método de subdivisão aplicado à geometria do ambiente virtual.

Como estamos na presença de um modelo de iluminação global, a qualidade da informação que traduz a relação geométrica entre superfícies (factor de forma) é de extrema importância. Alguns parâmetros poderão ser especificados a fim de melhorar a qualidade dos resultados obtidos.

Abstract

The global illumination models have given a special contribution in archiving the photo-realism goal in Image Synthesis, particularly the Global Illumination Model by Radiosity. This model marks a turning point in Image Synthesis, because it uses a well-known model from Thermodynamics to describe the energy transfer between surfaces. Light can be seen as a form of energy. Therefore, this model can be applied in Image Synthesis to model its behavior in a precise way.

Unfortunately radiosity algorithms requires huge amount of computational power, which were not easy to obtain when these algorithms first appeared. Today, personal computers have enough power, and specialized graphical hardware, to execute the computations needed in reasonable time. There are two main goals in this dissertation, the first it's to analyze the state of the art in Radiosity, the other is to build a prototype capable rendering scenes using these techniques.

To be able to apply this model it's necessary to prepare the scene. These preparations have a massive impact in the results' quality. One step is to apply this model is to subdivide the scene's geometry. The objective is to minimize the discretization error, because applying the Radiosity equation in the analytical form is impossible. The user will be able to specify in the prototype the subdivision method applied to the scene.

Global illumination models require information about relationship among geometric surfaces (form factor). Parameterization will be used to obtain quality information about these relationships.

Índice

1 Introdução 10

1.1 Motivação 10

1.2 Descrição e Contexto 12

1.3 Solução Apresentada 13

1.4 Principais Contribuições 14

2 Estado da Arte 16

2.1 Radiação 16

2.1.1 Intensidade 17

2.1.2 Frequência 17

2.1.3 Polarização 18

2.2 Radiosidade 18

2.3 Tipos de Modelos de Iluminação 19

2.4 Discretização da Equação de Radiosidade 19

2.5 Factores de Forma 20

2.5.1 A Analogia de Nusselt 21

2.5.2 O Semicubo 22

2.6 Solução Matricial de Radiosidade 24

2.7 Algoritmo de Radiosidade Progressiva 24

2.8 Subdivisão de Elementos 25

2.9 Aceleração por Hardware 26

2.9.1 Implementação do Semicubo por Hardware 27

2.9.2 Obtenção dos Factores de Fórmula por Parabolóide 27

2.10 Mapeamento de Radiosidade para Cor 29

3 Trabalho Relacionado 31

3.1 RadiosGL 31

3.2 Radvis: Radiosity Visualisation 32

3.3 RRV: Radiosity Renderer and Visualizer 34

4 Arquitectura e Implementação 36
4.1 Módulo de Renderização de Radiosidade 36
4.1.1 Obter informação sobre o ambiente virtual 38
4.1.2 Preencher as matrizes dos Factores de Forma Delta 41
4.1.3 Calcular os factores de forma entre elementos 42
4.1.4 Aplicar o Algoritmo de Radiosidade Progressiva 46
4.1.5 Interpolar os valores de radiosidade dos elementos 47
4.1.6 Registar os resultados da computação 48
4.2 Módulo de Exposição 49
4.3 Módulo de Visualização 50
4.4 Problemas e Soluções 51

5 Testes e Resultados 54
5.1 Qualidade 56
5.2 Desempenho 58

6 Conclusão 61
6.1 Trabalho Futuro 62
6.1.1 Profundidade de campo na visualização dos ambientes virtuais 62
6.1.2 Uso de BSP no acesso e armazenamento dos factores de forma 62
6.1.3 Obtenção dos Factores de Forma por Parabolóide 63
6.1.4 Implementação total em GPU (Graphics Processing Unit) 63

7 Anexos 65
7.1 Ângulo Sólido Diferencial 65
7.2 Norma de uma Função 66

8 Bibliografia 67

Índice de Figuras

Figura 1.1 - As caixas clássicas de Cornell onde se vê claramente a influência da cor das paredes nos objectos da cena e nas sombras dos mesmos. [5] 11
Figura 1.2 – Visão geral do funcionamento do sistema onde o utilizador pode antecipar o fim da computação da radiosidade se achar que o resultado actual é aceitável. 14
Figura 2.1 - Espectro de Radiação Electromagnética: Pode-se observar claramente que o espectro de luz visível está contido dentro deste espectro. [10] 17
Figura 2.2 - Correspondência entre a cor da luz e a sua frequência. 17
Figura 2.3 – Diagrama que sintetiza as variáveis que estão na base do cálculo dos factores de forma. [12] 20
Figura 2.4 - Na Analogia de Nusselt é colocado um hemisfério de raio unitário sobre o ponto que se pretende avaliar o factor de forma. [12] 21
Figura 2.5 - O factor forma é igual à área da projecção (A) a dividir pela área da base do hemisfério (B). [12] 21
Figura 2.6 – O factor de forma é igual em todas as intersecções das superfícies (A, B, C e ***Aj***), pois todas elas têm a mesma área quando projectadas na base do hemisfério. [9].....22
Figura 2.7 – O Semicubo é um modelo discreto da Analogia de Nusselt. Cada célula tem um Factor de Forma Delta pré-calculado. [12].....22
Figura 2.8 – Aproximação ao hemisfério da Analogia de Nusselt por um parabolóide. 28
Figura 2.9 – Imagem gerada com um ângulo de visão de 180 graus. [15] 29
Figura 2.10 – Gráfico de uma exponencial inversa semelhante à decomposição dos químicos presentes no filme de uma máquina fotográfica. O eixo das abcissas representa a quantidade de luz recebida e o eixo das ordenadas o químico restante. 30
Figura 2.11 – Gráfico que representa o impacto da quantidade de luz recebida pelo filme na intensidade da luz na imagem final. O eixo das abcissas representa a quantidade de luz recebida e o eixo das ordenadas a intensidade da luz. 30
Figura 3.1 – Onde se pode observar a combinação de cores ao longo da parede. 32
Figura 3.2 – Onde se pode observar as diferenças de intensidade ao longo da parede provocadas pela luz. 32

Figura 3.3 – O interface do Radvis permite ao utilizador parametrizar os diversos algoritmos de Radiosidade. 33
Figura 3.4 – Geometria inicial do ambiente virtual. 34
Figura 3.5 – Resultado da subdivisão de geometria para a aplicação do Algoritmo de Radiosidade. 34
Figura 3.6 – Resultado da computação do Algoritmo de Radiosidade sem interpolação. 35
Figura 3.7 – Resultado final, com interpolação nas superfícies. 35
Figura 4.1 – Etapas que o protótipo percorre para aplicar o Modelo de Iluminação Global por Radiosidade. 37
Figura 4.2 – Excerto do ficheiro COLLADA onde estão presentes as principais secções. 39
Figura 4.3 – Exemplo de uma secção de um ficheiro COLLADA onde está descrita a geometria dos objectos do ambiente virtual. 40
Figura 4.4 – Neste excerto de XML está descrito um ambiente virtual, bem como, as referências para os objectos que o compõem. 40
Figura 4.5 – Diagrama que mostra a relação entre a classe Scene e ColladaScene. 41
Figura 4.6 – Representação gráfica da matriz do topo do semicubo. Tons escuros equivalem a valores perto de 0, tons claros a valores perto de 1. 41
Figura 4.7 – Representação gráfica da matriz de uma das faces laterais do semicubo. 41
Figura 4.8 – Excerto de código que redirecciona o destino das imagens renderizadas pelo hardware gráfico para um buffer na VRAM. 44
Figura 4.9 – Diagrama da configuração usada pelo protótipo para renderizar imagens *offscreen*. 44
Figura 4.10 – Exemplo das Caixas Clássicas de Cornell, onde cada um dos elementos tem uma cor distinta. Esta cor serve de identificador único. Nenhum dos tons de vermelho na imagem é exactamente igual. 45
Figura 4.11 – Descrição da classe PatchViewFactorCache, usada para armazenar os factores de forma entre elementos. 46
Figura 4.12 – Diagrama onde as classes que compõem relação Fábrica/Produto de elementos está representada, bem como os métodos públicos de cada uma delas. 48
Figura 4.13 – Exemplo da secção de um ficheiro COLLADA onde estão descritas informações acerca da origem do ficheiro. 49
Figura 4.14 – Ambiente virtual que testa a dispersão da luz numa superfície sem interpolação entre elementos. 51
Figura 4.15 – Ambiente virtual que testa a dispersão da luz com alguns artefactos corrigidos. 51

Figura 4.16 – Ambiente virtual que testa a dispersão da luz, com interpolação entre elementos. 52
Figura 4.17 – Ambiente virtual que testa a transferência de cor entre superfícies. 53
Figura 4.18 – Influência que as paredes coloridas têm na superfície do chão. 53
Figura 4.19 – Ambiente virtual onde está presente um erro na distribuição da luz. 53
Figura 4.20 – Ambiente virtual sem problemas de *Z-Buffer*. 53
Figura 5.1 – Ambientes virtuais com iluminação por *hardware*. a) Parede. b) Xadrez. c) Caixas de Cornell. 55
Figura 5.2 – Ambientes virtuais com iluminação por radiosidade. a) Parede. b) Xadrez. c) Caixas de Cornell. 57
Figura 5.3 – Comparação dos tempos de renderização entre as duas estruturas de dados. 60
Figura 5.4 – Comparação da memória consumida durante a renderização, usando as duas estruturas de dados. 60

1 Introdução

Um dos objectivos desta dissertação é analisar e implementar o Modelo Global de Iluminação por Radiosidade. A radiosidade (na Computação Gráfica) é um algoritmo de iluminação global para ambientes virtuais compostos por superfícies difusas [1 pp. 479-481]. Este algoritmo foi inicialmente concebido para estudar a transferência de calor entre materiais. Mais tarde foi aplicado ao campo da computação gráfica, uma vez que a luz também é uma radiação electromagnética, à semelhança da radiação térmica.

Em paralelo ao objectivo anterior, pretende-se usar o hardware gráfico presente na maioria dos computadores pessoais para acelerar os cálculos necessários ao algoritmo de radiosidade.

1.1 Motivação

Um dos grandes objectivos da computação gráfica sempre foi produzir imagens de objectos e ambientes virtuais com um grande nível de foto-realismo. No entanto o hardware existente nem sempre possibilitou pôr em prática todas as teorias existentes. Quando é possível, o tempo para obter resultados nem sempre é compensatório, na medida em que muitos ambientes virtuais estão em constante evolução até atingirem o seu estado final ou simplesmente têm de ser renderizados em tempo-real. No entanto, a área não estagnou, pelo contrário, foram concebidos modelos mais simples como *Flat Shading* [2 pp. 370-371], *Gouraud Shading* [3] e *Phong Shading* [4 pp. 311-317]. Estes modelos produzem resultados satisfatórios e actualmente fazem parte integrante do hardware gráfico existente. Devido aos recentes avanços tecnológicos, muitos modelos e teorias podem ser postos em prática e os seus contributos analisados.

Esta dissertação tem como objectivo analisar e implementar um desses modelos, chamado Radiosidade. Este modelo de iluminação, ao contrário dos anteriormente referidos, não é baseado em algoritmos ad hoc nem em conhecimentos empíricos, o que adiciona uma nova camada de foto-realismo nunca antes observado. Na Figura 1.1 é possível observar pormenores como a transferência de cor das paredes para as caixas, e o impacto nas sombras

das mesmas. Estas transferências entre objectos ocorrem devido às características globais do algoritmo de radiosidade, visto este fazer parte da família dos modelos de iluminação global, ao contrário dos modelos de iluminação local (*Flat Shading* [2 pp. 370-371], *Gouraud Shading* [3] e *Phong Shading* [4 pp. 311-317]). Estes modelos têm em conta a influência que outros objectos podem provocar na iluminação do objecto em estudo, o que harmoniza a sua aparência com o ambiente onde se encontram.

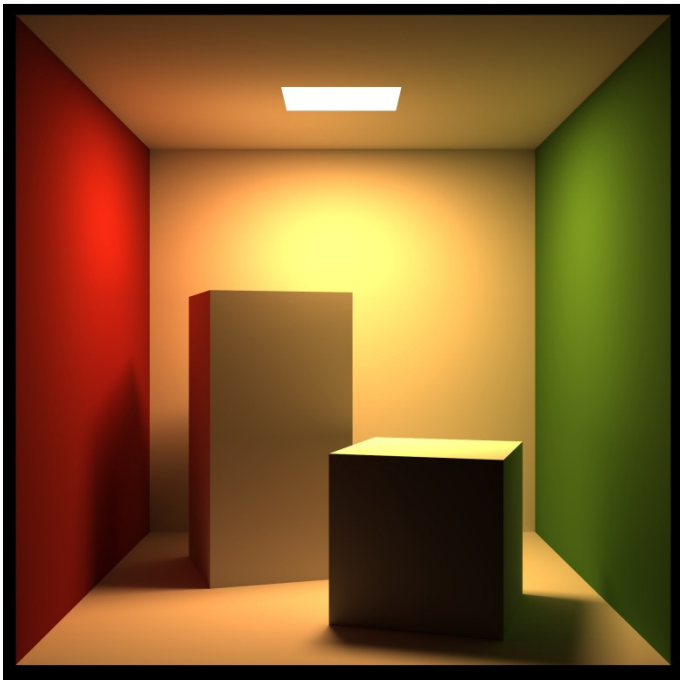

Figura 1.1 - As caixas clássicas de Cornell onde se vê claramente a influência da cor das paredes nos objectos da cena e nas sombras dos mesmos. [5]

Um dos maiores problemas na aplicação do algoritmo de radiosidade consiste em obter os factores de forma entre as superfícies dos objectos, ou secções de superfície designadas ao longo desta dissertação como "elementos". Estes factores de forma fazem parte da equação clássica de radiosidade e caracterizam a relação geométrica entre duas superfícies lambertianas [1 pp. 479-481], isto é, traduz a fracção de energia que parte de uma superfície e chega a outra. O cálculo deste coeficiente pode ser feito de diversas formas, no entanto uma das formas mais simples baseia-se na Analogia de Nusselt (Figura 2.4 e Figura 2.5). É exactamente nesta tarefa que a aceleração por hardware tem maior impacto, visto que o cálculo dos factores de forma tem de ser feito entre todos os pares de superfícies do ambiente virtual.

Um factor muito importante na obtenção de um ambiente virtual foto-realista reside na técnica de subdivisão de superfícies usada. A equação clássica de radiosidade é definida num domínio contínuo. Porém, quando aplicada à computação gráfica, é necessário discretizar a equação, transformando o integral num somatório, uma vez que os computadores lidam apenas com domínios de valores discretos. Ao proceder a esta conversão estamos obviamente a incorrer em erros de discretização. Para os atenuar, a geometria presente tem de ser subdividida. No entanto, se a subdividirmos indiscriminadamente estamos a despender recursos computacionais em áreas que poderiam não necessitar dessa subdivisão. Existem algumas formas de garantir que esse desperdício de recursos não ocorre. Uma das mais simples consiste em subdividir manualmente (recorrendo a editores gráficos) a geometria nas zonas onde existe uma transição de intensidade brusca, isto é, nas fronteiras das sombras. Este método pode não ser aplicável em todas as situações, especialmente em ambientes virtuais muito complexos. No entanto, existem formas de resolver este problema recorrendo a algoritmos de subdivisão de superfícies [6].

1.2 Descrição e Contexto

O Modelo de Iluminação Global por Radiosidade aplicado a um ambiente virtual permite gerar imagens onde o nível de realismo é um ponto forte, especialmente na interacção entre a luz e os objectos presentes no ambiente.

Uma das áreas que beneficia grandemente com as propriedades deste modelo é a da Arquitectura. A Arquitectura e a luz são conceitos interdependentes. Le Corbusier, famoso arquitecto e urbanista, citado por César Portela, chegou mesmo a dizer que *"architecture is the wise, correct and magnificent play of volumes collected together under the light"* [7]. A interacção entre a luz e a arquitectura é inevitável, por vezes é consciente, outras não. No entanto, a resultante dessa interacção influencia a percepção que temos do contexto espacial, pode torná-lo agradável ou desagradável, sublime ou misterioso, pode dar a sensação de um espaço grande ou a ilusão de um pequeno. Todas estas influências percepcionais tornam os espaços mais confortáveis e habitáveis, podem fazer sobressair certas características e ocultar outras. A capacidade de poder observar o comportamento da luz num ambiente virtual é de extrema importância para poder corrigir possíveis problemas ou simplesmente testar novas ideias.

Outra vantagem consiste em mostrar, a potenciais clientes, estruturas ou interiores de edifícios que ainda não foram construídos. Apesar de existirem muitas ferramentas gráficas

para arquitectura, as imagens que estas produzem normalmente são de difícil compreensão para pessoas que não tenham formação na área. Por outro lado as imagens foto-realistas são universais e podem servir de base para o cliente exprimir melhor a sua opinião ou para o arquitecto fundamentar algumas das suas decisões e explicar melhor as suas ideias.

1.3 Solução Apresentada

Um dos pontos fortes do algoritmo de radiosidade é que só necessita de ser aplicado uma vez por cada ambiente virtual, assumindo que o ambiente virtual não sofre alterações durante a sua visualização. Isto acontece porque as transferências de energia entre superfícies dependem apenas do factor de forma e da reflectividade de ambas as superfícies. A posição de observação não influencia o resultado do algoritmo.

Para aplicar o algoritmo de radiosidade é necessário ter informação sobre o ambiente virtual. Esta é passada ao sistema por via de um ficheiro de entrada com os seguintes dados:

- Objectos:
    - Geometria.
    - Posicionamento.
    - Materiais:
        - Reflectividade (cor).
        - Emissividade.
        - Texturas.
        - *Shaders* [8].

Para poder guiar o sistema na aplicação do algoritmo de radiosidade, são usados os seguintes parâmetros:

- Número de iterações máximas permitidas.
- Regularidade de geração de resultados intermédios.
- Distância do plano de projecção.
- Distância do plano de corte.
- Resolução interna de texturas auxiliares.

À medida que a radiosidade do ambiente virtual é calculada, é possível observar a evolução do algoritmo. Num intervalo regular de iterações do Algoritmo de Radiosidade Progressiva, os resultados intermédios serão guardados e poderão ser visualizados. Em qualquer altura o algoritmo pode ser interrompido e os resultados intermédios da computação poderão ser utilizados como os finais (Figura 1.2).

O sistema produz um ficheiro com os valores de radiosidade que obteve até ao momento. Este é usado como ficheiro de entrada a um visualizador, para que se possa navegar pelo ambiente virtual, ou servir de ficheiro de entrada a outra aplicação que necessite dos dados.

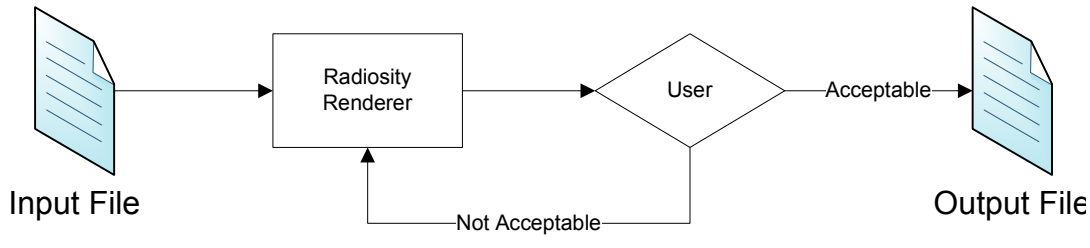

Figura 1.2 – Visão geral do funcionamento do sistema onde o utilizador pode antecipar o fim da computação da radiosidade se achar que o resultado actual é aceitável.

1.4 Principais Contribuições

No âmbito da análise do estado da arte do Modelo de Iluminação Global por Radiosidade, foi concebido um protótipo capaz de renderizar e apresentar um ambiente virtual recorrendo a técnicas e algoritmos usados por este modelo.

O sistema é composto por dois módulos: o renderizador de radiosidade e o visualizador de ambientes virtuais.

O módulo de renderização de radiosidade permite:

- Especificar a resolução do Semicubo no cálculo dos factores de forma entre superfícies;
- Especificar a condição de paragem, com base no número de iterações do Algoritmo de Radiosidade Progressiva;
- Calcular a intensidade de todas as superfícies usando o Algoritmo de Radiosidade Progressiva com aceleração por hardware.

O módulo de visualização permite:

- Visualizar o ambiente virtual de diferentes formas, entre elas os tradicionais *wireframe* e *flat shading*;
- Navegar pelo ambiente virtual usando o rato e o teclado;
- Observar os resultados das computações intermédias do Algoritmo de Radiosidade Progressiva.

O ficheiro de saída (que contém o resultado das computações do algoritmo de radiosidade) poderá ser usado por outros sistemas, sem que estes tenham de calcular os valores de radiosidade.

Para além das contribuições anteriores existe um aspecto relativo ao protótipo que é importante referir. O subsistema responsável pelo cálculo dos factores de forma, utilizou hardware gráfico comum para simplificar e acelerar a sua obtenção.

2 Estado da Arte

Para melhor compreender o Modelo de Iluminação Global por Radiosidade é necessário analisar os fenómenos físicos associados que ocorrem na natureza. Outro objectivo desta secção é mostrar as fórmulas matemáticas que modelam estes fenómenos físicos, bem como os algoritmos que recorrem a estas para gerar ambientes virtuais credíveis [9].

2.1 Radiação

Para percebemos melhor o modelo de iluminação por radiosidade é necessário observar o fenómeno físico que está por trás. Todos os corpos têm a capacidade de emitir radiação, mas nem todos emitem radiação da mesma forma ou do mesmo tipo.

A radiação é o processo em que um corpo emite energia e esta espalha-se pelo seu ambiente. Esta energia pode ser absorvida por outros corpos que a voltarão a emitir, sob a forma do mesmo tipo de radiação ou não, consoante as suas próprias características.

Existem muitos tipos de radiação provocadas por diversas fontes, mas a mais relevante para compreender o modelo de iluminação por radiosidade é a radiação electromagnética. Esta radiação é importante, pois o seu espectro comporta o espectro de luz visível (Figura 2.1).

Devido ao espectro de luz visível estar contido no electromagnético, podemos usar as suas propriedades para melhor modelar as características da luz visível.

As três propriedades principais da luz são a intensidade, a frequência e a polarização. Estas propriedades serão descritas nas próximas subsecções.

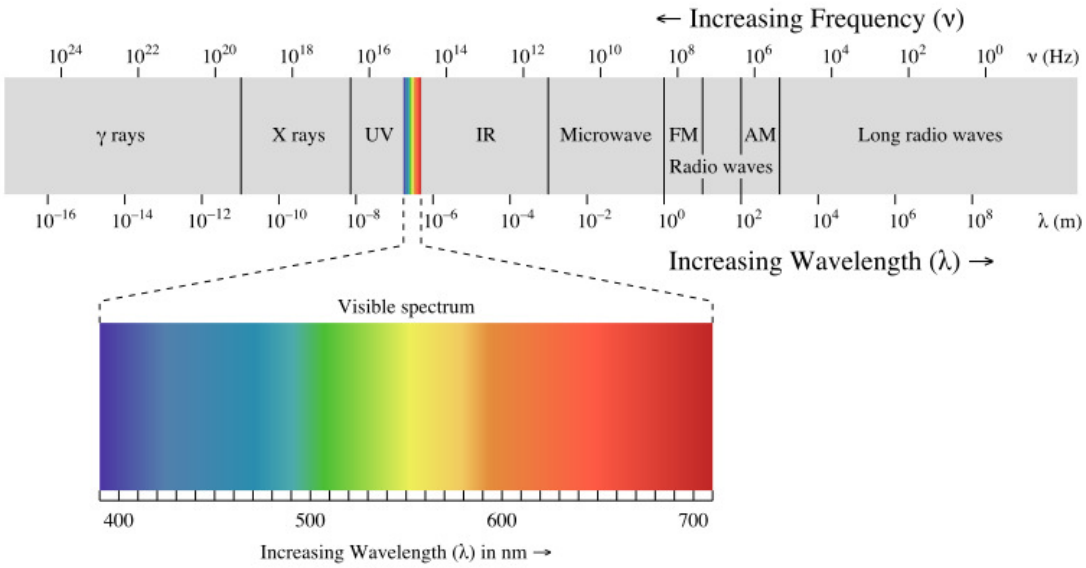

Figura 2.1 - Espectro de Radiação Electromagnética: Pode-se observar claramente que o espectro de luz visível está contido dentro deste espectro. [10]

2.1.1 Intensidade

A intensidade mede o fluxo de energia ao longo do tempo, sendo a potência de uma fonte de intensidade dada pela expressão (2.1), onde P é a potência, dA é um infinitésimo de área e I é a intensidade.

$$P = \int I \, dA \qquad (2.1)$$

A intensidade é muitas vezes referida como radiância (especialmente na astronomia), sendo esta uma propriedade muito importante no estudo da radiosidade.

2.1.2 Frequência

A frequência da luz diz respeito à sua natureza ondulatória. É a frequência da luz que dita a cor que vemos, como se pode observar na Figura 2.2.

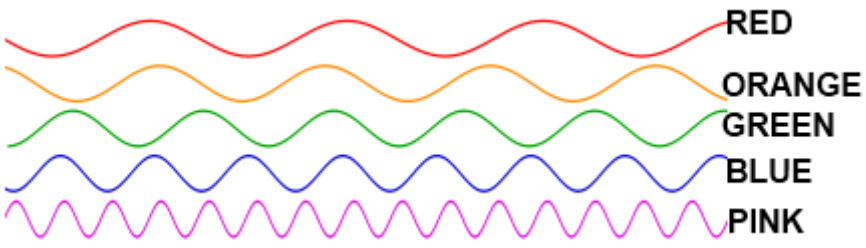

Figura 2.2 - Correspondência entre a cor da luz e a sua frequência.

A frequência é dada pela expressão (2.2), onde f é a frequência em Hertz e T o período da onda (tempo que demora a onda a repetir-se).

$$f = \frac{1}{T} \qquad \textbf{(2.2)}$$

2.1.3 Polarização

Como a luz tem uma componente de onda, esta também pode ter polarizações. A polarização dá-se devido à variação do vector do campo eléctrico ao longo do tempo, o que produz uma onda tridimensional, em oposição a ondas planas como as aquáticas ou sonoras.

A iluminação artificial emite radiação incoerente, isto é, não se pode classificar o tipo de polarização. Isto ocorre porque este tipo de luz é composto por diversas frequências com características distintas.

A polarização da luz pode ser filtrada recorrendo a um filtro polarizador, o que remove certas reflexões presentes nos objectos.

A polarização da luz incidente é uma componente muito importante para o realismo de uma cena, no entanto é quase sempre ignorada por motivos de simplificação de algoritmos.

2.2 Radiosidade

Como foi referido anteriormente, a radiosidade é um algoritmo de iluminação global para cenas compostas por superfícies difusas. Este algoritmo foi inicialmente concebido para estudar a transferência de calor entre materiais sendo mais tarde aplicado ao campo da computação gráfica, devido ao facto de a luz ser também uma radiação electromagnética à semelhança da radiação térmica. A radiosidade também é uma propriedade física que caracteriza a energia que abandona uma superfície por unidade de área. É caracterizada pela expressão (2.3), onde B é a radiosidade, L_o a radiância, θ o ângulo de incidência a partir do zénite (ver Anexo 7.1) e Ω o hemisfério superior.

$$B = \int_{\Omega} L_o \cos\theta \, d\omega \qquad (2.3)$$

Estas transferências de energia entre superfícies dependem apenas das relações geométricas entre os elementos (*patches*), como está descrito na secção 2.5, logo, a posição de observação não influencia o resultado do algoritmo. Este facto faz com que os resultados

obtidos sejam independentes do ponto de observação, o que é muito vantajoso pois possibilita navegar no ambiente virtual sem se ter de recalcular novamente a solução de radiosidade.

2.3 Tipos de Modelos de Iluminação

Todas as superfícies físicas são iluminadas por dois tipos de luz: directa e indirecta. A luz directa é aquela que chega à superfície directamente, descrevendo o caminho mais curto entre uma fonte e a superfície. Os modelos clássicos de iluminação *Flat Shading* [2 pp. 370-371], *Gouraud Shading* [3] e *Phong Shading* [4 pp. 311-317] modelam bem este fenómeno e o hardware actualmente existente está bastante optimizado para lidar com este tipo de iluminação. Como estes modelos não têm em conta a luz indirecta (luz que é reflectida de outros objectos) é necessário introduzir uma componente de luz extra chamada de luz ambiente. A intensidade desta luz é escolhida empiricamente, e como normalmente é aplicada a todos os objectos da cena de uma forma uniforme, deixa muito a desejar em termos de realismo. No entanto, os modelos de iluminação global têm em conta não só a luz directa como a luz indirecta. Devido a esse facto, não necessitam de atalhos para representar os efeitos da luz ambiente (luz indirecta), obtendo por isso muito melhores resultados ao nível do realismo.

2.4 Discretização da Equação de Radiosidade

Assumindo que todas as superfícies são perfeitamente difusas [11], a radiação de uma determinada superfície é dada pela expressão (2.4), onde $B(x)$ representa o valor de radiosidade do ponto x do domínio S (espaço do ambiente virtual), $E(x)$ a emissividade no ponto x, $B(x')$ o valor de radiosidade do ponto x' do domínio S, $G(x, x')$ quantifica a relação geométrica entre os dois pontos e a visibilidade, representando dA' a área diferencial no ponto x, isto é, $dx\ dy$.

$$B(x) = E(x) + \rho(x) \int_S B(x')G(x,x')dA' \qquad (2.4)$$

Como a equação (2.4) contém um integral, torna-se muito difícil achar a solução computacionalmente. Para tornar esta expressão mais acessível a meios informáticos é necessário proceder à sua discretização. É aí que a equação clássica da radiosidade (2.5) surge, onde B_i é a radiosidade da superfície i, E_i é a emissividade da superfície i, ρ_i é a reflectividade da superfície i, B_j é a radiosidade da superfície j e F_{ij} representa o factor de

forma (explicado na secção 2.5) entre a superfície i e a superfície j, sendo n o número de superfícies.

$$B_i = E_i + \rho_i \sum_{j=1}^{n} B_j F_{ij} \qquad (2.5)$$

Ao procedermos à discretização da equação (2.4) incorremos obrigatoriamente em erros de aproximação. Estes erros são minimizados em fases mais avançadas do algoritmo, recorrendo, por exemplo, a técnicas de interpolação.

Se aplicarmos a fórmula a um conjunto de superfícies, iremos obter um sistema de equações cuja representação matricial é dada pela expressão (2.6).

$$\begin{bmatrix} 1-\rho_1 F_{11} & -\rho_1 F_{12} & \cdots & -\rho_1 F_{1n} \\ -\rho_2 F_{21} & 1-\rho_2 F_{22} & \cdots & -\rho_2 F_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -\rho_n F_{n1} & -\rho_n F_{n2} & \cdots & 1-\rho_n F_{nn} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{bmatrix} \qquad (2.6)$$

Existem duas formas de calcular a radiosidade das superfícies, que serão abordadas com mais detalhe na secção 2.6 e 2.7.

2.5 Factores de Forma

O factor de forma é uma componente da equação clássica de radiosidade (como vimos na secção 2.4). Esta componente pode assumir significados mais ou menos intuitivos que traduzem a relação geométrica entre duas superfícies, como se pode observar na Figura 2.3. O factor de forma não depende da emissividade ou da reflectividade das superfícies, mas depende da visibilidade entre elas. Em suma, traduz a fracção de energia que parte de uma superfície e chega a outra.

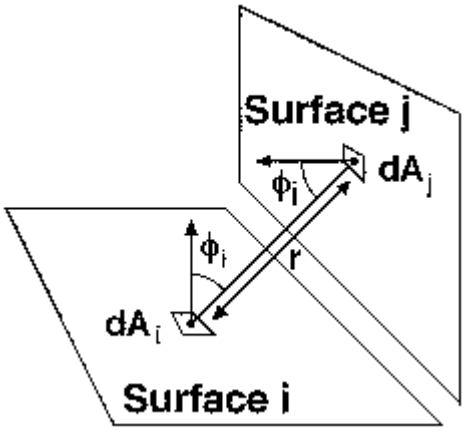

Figura 2.3 – Diagrama que sintetiza as variáveis que estão na base do cálculo dos factores de forma. [12]

O modelo matemático que caracteriza esta transacção de energia é dado pela equação (2.7), em que dA_i e dA_j são as áreas diferenciais das superfícies i e j respectivamente, r o vector entre dA_i e dA_j, e ϕ_i e ϕ_j representam o ângulo entre as normais das superfícies i e j e o vector r.

$$F_{ij} = \frac{1}{A_i}\int_{A_i}\int_{A_j}\frac{\cos\phi_i \cos\phi_j}{\pi|r|^2}dA_i dA_j \qquad (2.7)$$

No entanto existem outros métodos para calcular os factores de forma entre superfícies. Alguns deles serão apresentados de seguida.

2.5.1 A Analogia de Nusselt

Este método para obter o factor de forma é um dos mais simples e intuitivos que existem. Como o factor de forma é um problema puramente geométrico, nada como usar uma solução geométrica para o obter.

Na Analogia de Nusselt é colocado um hemisfério de raio unitário sobre o ponto que se pretende avaliar o factor de forma, como se pode observar pela Figura 2.4. Projectando as superfícies em redor radialmente no hemisfério e de seguida fazendo uma projecção ortogonal do hemisfério na sua base, podemos comparar a área resultante da projecção e a área da base do hemisfério, como demonstra a Figura 2.5. O factor de forma é igual à área da projecção a dividir pela área da base do hemisfério, o que torna a obtenção do factor de forma simples e intuitiva.

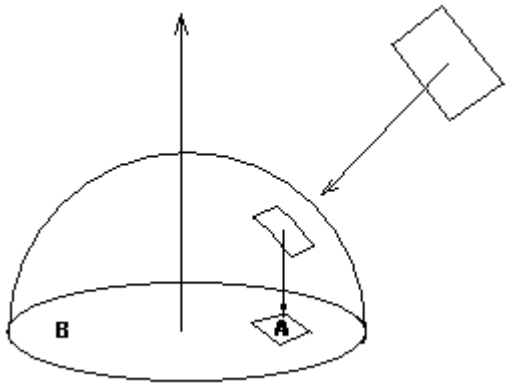

Figura 2.4 - Na Analogia de Nusselt é colocado um hemisfério de raio unitário sobre o ponto que se pretende avaliar o factor de forma. [12]

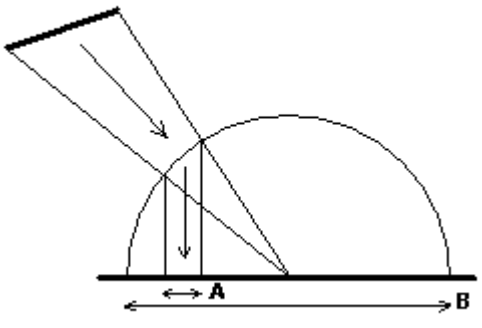

Figura 2.5 - O factor forma é igual à área da projecção (A) a dividir pela área da base do hemisfério (B). [12]

2.5.2 O Semicubo

A Analogia de Nusselt mostra que se dois elementos projectados radialmente no hemisfério cobrirem a mesma área, têm o mesmo factor de forma. Logo, se um elemento for projectado radialmente numa superfície intermédia, o factor de forma será o mesmo que o desta superfície, como ilustra a Figura 2.6.

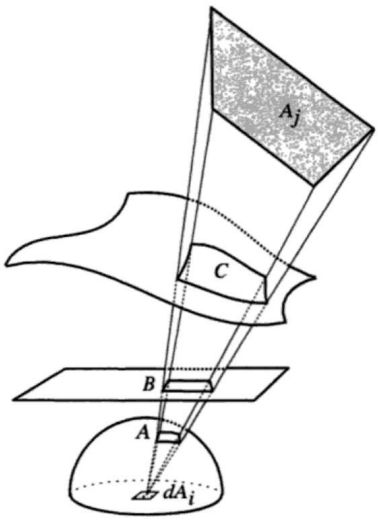

Figura 2.6 – O factor de forma é igual em todas as intersecções das superfícies (A, B, C e A_j), pois todas elas têm a mesma área quando projectadas na base do hemisfério. [9]

O semicubo unitário (i.e., com altura de 1 unidade e a face do topo com 2 x 2 unidades), à semelhança do hemisfério da Analogia de Nusselt, é colocado na área diferencial (ou ponto) onde se pretende obter o factor de forma, como se pode observar na Figura 2.7.

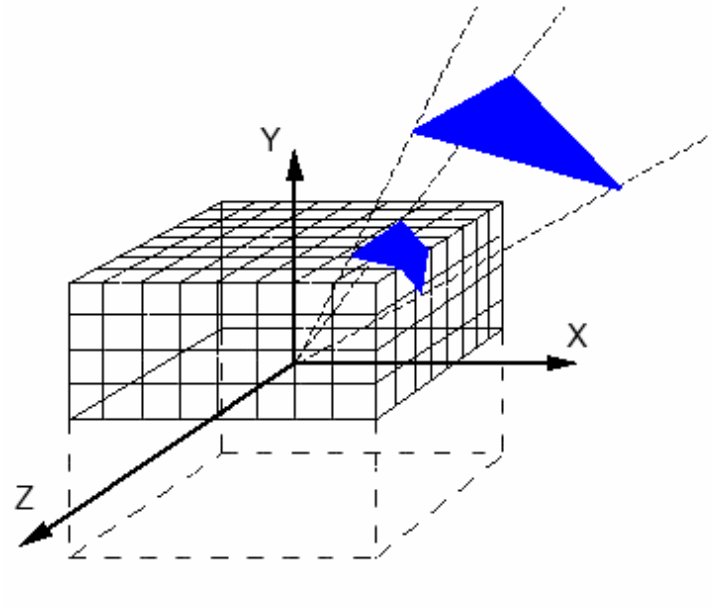

Figura 2.7 – O Semicubo é um modelo discreto da Analogia de Nusselt. Cada célula tem um Factor de Forma Delta pré-calculado. [12]

As faces do semicubo são subdivididas numa grelha composta por pequenas células. Cada célula define uma direcção (até ao centro do semicubo) e um ângulo sólido (ver Anexo 7.1). Com esta informação é possível calcular um factor de forma delta, ΔF, para todas as células. De notar que só é necessário calcular um oitavo dos factores de forma delta devido à simetria do semicubo (se dividirmos as faces do cubo em 4 partes iguais, só é necessário uma das partes da face de cima e duas partes de uma das faces laterais).

Para calcular um factor de forma delta de uma célula do topo do semicubo usa-se a fórmula (2.8), onde ΔA_i é a área diferencial da célula i.

$$\Delta F_{dA_i} = \frac{1}{\pi\sqrt{x_i^2 + z_i^2 + 1}}\Delta A_i \qquad (2.8)$$

O cálculo do factor de forma delta das células das faces laterais do semicubo é feito recorrendo à fórmula (2.9), em que ΔA_j é a área diferencial da célula j.

$$\Delta F_{dA_j} = \frac{1}{\pi\sqrt{y_j^2 + z_j^2 + 1}}\Delta A_j \qquad (2.9)$$

Cada célula define um plano de projecção que é usado para projectar os elementos da cena. Para solucionar o problema de visibilidade pode-se usar algoritmos de síntese de imagem como o *Z-Buffer* [1 pp. 451-454]. A maior diferença entre a renderização tradicional e a renderização para o semicubo prende-se com o facto de os elementos terem de ser identificados, para mais tarde se proceder à sua quantificação. Existem muitas formas de manter a identidade de um elemento. Uma das mais simples é usar os canais de cor para armazenar o identificador do elemento (Figura 4.10).

Depois de projectar os elementos na grelha de células basta somar os factores de forma delta que foram cobertos pela projecção, conforme mostra a equação (2.10), em que q representa o número de células cobertas pelo elemento j.

$$F_{ij} = \sum_{q \epsilon j} \Delta F_q \qquad (2.10)$$

Sendo o semicubo uma discretização do hemisfério da Analogia de Nusselt é normal a acumulação de erros, que resultam devido ao *aliasing* que ocorre quando os elementos ocupam poucas células. O aumento da resolução do semicubo reduz o problema, mas torna-se altamente ineficiente, uma vez que produz um maior o esforço de computação indiscriminadamente.

Existem algumas variantes do semicubo que atenuam estes problemas, mas cuja complexidade pode ser dissuasora. Apesar disto, o semicubo ainda é uma das melhores formas de calcular os factores de forma computacionalmente, uma vez que é possível usar hardware gráfico para acelerar o algoritmo sem grande dificuldade (como se pode observar na secção 2.9.1).

2.6 Solução Matricial de Radiosidade

Este método resolve a matriz com todas as superfícies presentes na cena. Como seria de esperar, esta solução requer calcular o factor de forma entre todas as superfícies e depois resolver a matriz em ordem aos valores de B a fim de calcular a radiosidade de cada uma das superfícies. Para além de ser computacionalmente muito dispendioso, também requer grandes quantidades de memória. Não é invulgar termos ambientes virtuais compostos por milhares de superfícies, especialmente quando previamente são aplicadas técnicas de tesselização aos objectos.

Algoritmos como Eliminação de Gauss e as suas variantes podem ser aplicados a este problema. No entanto apresentam complexidade na ordem de $O(n^3)$, o que limita a sua aplicação apenas a matrizes de pequena dimensão.

É também possível usar métodos iterativos de resolução de matrizes. Estes algoritmos começam com valores arbitrários, sendo estes refinados a cada iteração. Aplicando operações simples a estes valores, espera-se que tendam para a solução final.

2.7 Algoritmo de Radiosidade Progressiva

O Algoritmo de Radiosidade Progressiva é muito semelhante ao algoritmo Southwell da família de algoritmos de relaxação matricial [13]. À semelhança deste, também é um algoritmo iterativo que, por cada iteração, se aproxima da solução final (valores finais de B usando a Matriz Total de Radiosidade). Visto que se trata de um método iterativo, cada iteração produz um determinado resultado que pode ser satisfatório, não sendo assim necessário chegar à solução final. Ao contrário do método por Matriz Total de Radiosidade, este algoritmo não necessita de calcular N^2 factores de forma (onde N é o número de superfícies).

Este algoritmo pode ser resumido da seguinte forma. Todas as superfícies do ambiente virtual têm uma variável B_i, que representa o valor da radiosidade calculado até ao momento, e a variável ΔB_i, que guarda o valor da radiosidade que ainda não foi emitido. Durante uma iteração, a superfície com mais radiosidade por emitir é seleccionada. Todas as outras superfícies j podem receber parte da radiosidade a ser emitida. Este valor é calculado recorrendo aos factores de forma entre superfícies i e as superfícies j. O valor obtido é adicionado a B_j e a ΔB_j, pois a superfície j ainda não emitiu este valor de radiosidade recebido. No fim de iterar sobre todas as superfícies j, ΔB_i é posto a zero na medida em que toda a radiosidade da superfície foi emitida.

O pseudocódigo para este algoritmo é o seguinte:

```
Para todo o i:
        -A radiosidade da superfície i é igual à sua emissividade.
        -A radiosidade não emitida da superfície i é igual à sua emissividade.

Enquanto existir radiosidade não emitida:
        -Seleccionar a superfície i tal que esta tenha o maior valor de radiosidade não emitida
por área.

        Para cada uma das outras superfícies j:
                -Calcular o factor de forma entre a superfície i e a j.
                -Calcular o valor de energia transmitida da superfície i para a j.
                -Somar esse valor à radiosidade não emitida da superfície j.
                -Somar também o valor de energia transmitida à radiosidade da superfície j.

        -Colocar o valor de radiosidade não emitida da superfície i a zero.
        -Mostrar a imagem usando os valores de radiosidade obtidos.
```

Como podemos observar pelo algoritmo anterior, o custo de armazenagem dos valores dos factores de forma é exactamente $N - 1$ (se não considerarmos mecanismos de *cache* entre iterações).

2.8 Subdivisão de Elementos

Na secção 2.4 foi descrito a forma de discretiza a função de radiosidade. Nesta secção são abordadas formas de subdividir a geometria da cena a fim de minimizar o erro de discretização, sendo necessário primeiramente calcular este erro.

Analiticamente o erro é dado pela fórmula (2.11), em que $\hat{B}(x)$ é a aproximação (função discretizada) de $B(x)$. Como $B(x)$ não é conhecido, $\varepsilon(x)$ tem de ser estimado.

$$\varepsilon(x) = B(x) - \hat{B}(x) \qquad (2.11)$$

Para estimar o erro local de um elemento podemos começar pela aproximação inicial presente na expressão (2.12), onde $N_i(x)$ são as funções base de ordem k que caracteriza a transferência de energia entre as superfícies.

$$\hat{B}(x) = \sum_{i=1}^{n} B_i N_i(x) \qquad (2.12)$$

Compare-se esta função com uma de ordem superior (2.13), onde $\widetilde{N}_i(x)$ são as funções base de ordem $k+1$. O estimador do erro é obtido subtraindo a expressão (2.13) à expressão (2.12), como se pode observar na equação (2.14).

$$\tilde{B}(x) = \sum_{i=1}^{n} B_i \widetilde{N}_i(x) \qquad (2.13)$$

$$\hat{\varepsilon}(x) = \hat{B}(x) - \tilde{B}(x) \qquad (2.14)$$

Para decidir se a superfície deve ser subdividida podemos obter a magnitude do erro a partir da norma da função $\hat{\varepsilon}(x)$ (ver Anexo 7.2). Considerando alguns pontos da superfície em estudo, esta é comparada com um valor de referência e, se for superior a este, a superfície é subdividida.

2.9 Aceleração por Hardware

A aplicação do modelo de radiosidade num ambiente virtual é um processo muito dispendioso computacionalmente. Para atenuar os custos inerentes a estes cálculos desenvolveram-se várias técnicas que tiram partido de hardware especializado. Este hardware começa a ser comum em muitos computadores pessoais, o que deu origem a APIs (*Application Programming Interfaces*) como o CUDA da NVidea e o OpenCL [14].

Também é possível tirar partido do hardware sem ser necessário recorrer a soluções extremas como o CUDA ou o OpenCL [14], que muitas vezes requerem alterações nas estruturas de dados para tirar partido deste tipo de hardware. Os processadores dos computadores pessoais contêm cada vez mais cores e extensões, como o SSE2 (*Streaming*

SIMD Extensions 2), que permitem usar algoritmos existentes sem grandes alterações na estrutura de dados e mesmo assim obter desempenhos muito superiores em relação a uma solução sequencial.

2.9.1 Implementação do Semicubo por Hardware

Na secção 2.5, onde se discute um dos métodos possíveis para obter os factores de forma, os elementos do ambiente virtual são projectados nas superfícies do semicubo. Estas projecções são muito dispendiosas computacionalmente. Uma forma de as obter a um custo muito reduzido é colocar o plano de projecção em cada uma das faces do semicubo e, de seguida, renderizar a imagem usando OpenGL ou DirectX. Visto que a função principal das placas gráficas é renderizar ambientes tridimensionais para representações bidimensionais, fazendo assim uma projecção efectiva no plano de projecção, o custo desta operação é bastante reduzido.

No entanto é necessário aceder a estas imagens para que os *texels* possam ser avaliados. O acesso a essa imagem em OpenGL faz-se usando a função **glReadPixels**. Esta função copia o conteúdo do *frame buffer* (última imagem renderizada) da VRAM (memória volátil da placa gráfica) para a RAM (memória volátil de uso generalista). Trata-se de uma operação lenta, uma vez que o hardware gráfico não está optimizado para fazer outputs para a RAM, mas sim directamente para o monitor. Para contornar esta situação é possível aceder a VRAM directamente a partir do CUDA ou OpenCL [14]. No entanto, a solução de radiosidade terá de ser implementada nestas arquitecturas, para que os cálculos se façam no hardware gráfico e não seja preciso disponibilizar o *frame buffer* ao CPU da máquina.

2.9.2 Obtenção dos Factores de Fórmula por Parabolóide

Ao analisar com atenção a Analogia de Nusselt (secção 2.5.1) é tentador usar um hemisfério como superfície de projecção para obter os factores de forma, o que é possível mas incorre-se em erros de amostragem devido às características de amostragem inconstantes de superfícies esféricas [15]. As projecções no semicubo não sofrem de tantos erros de amostragem, mas é necessário renderizar as cinco faces visíveis do semicubo. A projecção

para um parabolóide (Figura 2.8) possui erros de amostragem inferiores aos do hemisfério mas superiores aos do semicubo. É, portanto, uma solução de compromisso entre velocidade e qualidade. Na radiosidade em tempo-real, pode ser um factor chave, pois reduz drasticamente os custos computacionais relacionados com a obtenção dos Factores de forma.

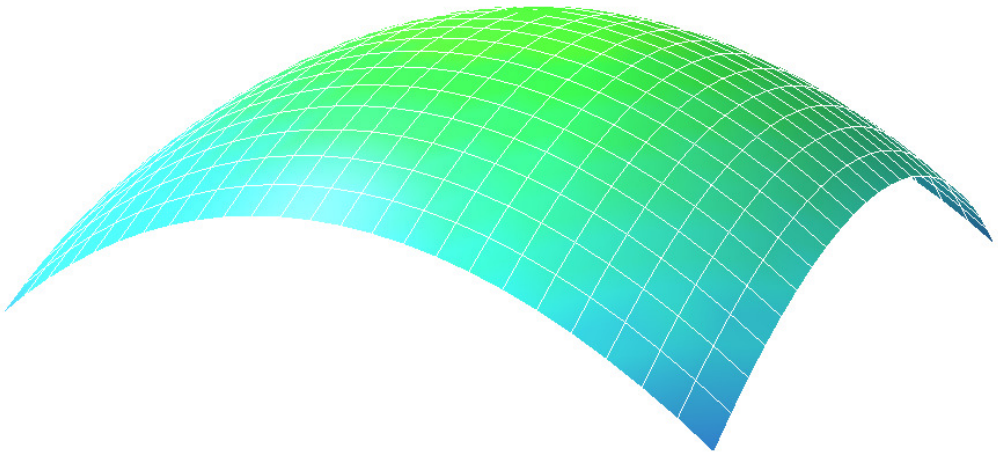

Figura 2.8 – Aproximação ao hemisfério da Analogia de Nusselt por um parabolóide.

A projecção para um parabolóide permite renderizar uma imagem com um campo de visão de 180 graus. Ao coloca-lo sobre a superfície da área diferencial é necessário fazer apenas uma projecção, uma vez que o parabolóide cobre todos os possíveis caminhos por onde a luz pode chegar à área diferencial.

$$f(x,y) = \frac{1}{2} - \frac{1}{2}(x^2 + y^2) \quad para \quad x^2 + y^2 \leq 1 \qquad (2.15)$$

O modelo matemático usado para fazer as projecções é descrito na equação (2.15). É normalmente implementado num *vertex shader*, que transforma os vértices da geometria à medida que o hardware gráfico os renderiza. A imagem final pode ser observada na Figura 2.9. Contabilizando os píxeis em conjugação com a sua posição na imagem é possível obter uma aproximação dos Factores de forma em relação a uma determinada área diferencial, tal como acontece no método do semicubo (secção 2.5.2).

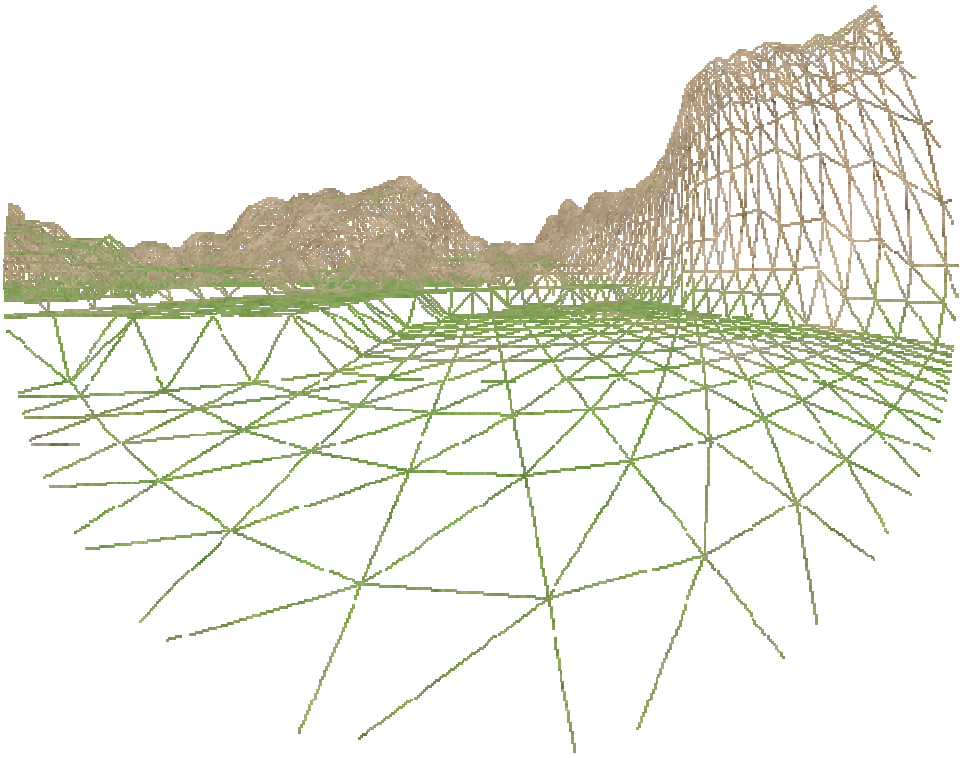

Figura 2.9 – Imagem gerada com um ângulo de visão de 180 graus. [15]

2.10 Mapeamento de Radiosidade para Cor

Quando o cálculo da radiosidade de um ambiente virtual termina, obtemos o equilíbrio do valor energético emitido por elemento. No entanto, esse valor pode variar entre zero e infinito, o que impede a sua utilização directa no hardware existente, pois este usa valores normalizados entre zero e um (ou 0 e 255) por canal de cor.

Mapear os valores de radiosidade linearmente nos canais de cor não é aconselhado, uma vez que o sistema visual humano não percepciona as intensidades de forma linear, logo muito detalhe da imagem é perdido.

Existem muitas formas de mapear os valores de radiosidade [16] [17]. A maioria baseia-se em conhecimento da área da fotometria, particularmente relacionadas com fotografia.

Quando uma fotografia é tirada, o obturador expõe o filme à luz durante breves momentos. O filme, por sua vez, contém químicos sensíveis à luz que se decompõem e registam a imagem. A taxa de decomposição dos químicos usados é semelhante a uma exponencial inversa, em que o eixo das abcissas representa a quantidade de luz recebida e o das ordenadas

a quantidade de químico restante, como se pode observar na Figura 2.10. Quando o filme é revelado, obtém-se o seu negativo, onde as zonas mais transparentes são aquelas onde a luz destruiu mais quantidade de moléculas dos químicos. Assim, através da inversão do gráfico da Figura 2.10, obtém-se o gráfico da Figura 2.11, onde a relação entre tempo de exposição à luz e intensidade é dada pela equação (2.16) [18].

$$intensity = 1 - e^{-light} \qquad (2.16)$$

O controlo da exposição do filme à luz faz-se com base no tempo que o obturador está aberto e o tamanho da abertura do diafragma. Para simplificar este modelo pode-se combinar estas duas variáveis e obter a variável K. Assim, a variável K controla o tempo de exposição. Multiplicando esta variável pela variável $light$ da equação (2.16) obtém-se a equação (2.17). Esta pode ser usada para converter radiosidade em cor com base num valor K arbitrado.

$$intensity = 1 - e^{-light \times K} \qquad (2.17)$$

Quando K assume valores pequenos em relação aos valores da variável $light$, as imagens obtidas são escuras. Quando este assume valores grandes, as imagens obtidas são mais claras. Esta variável, ao ser ajustada, dá o controlo equivalente ao tempo de exposição que as máquinas fotográficas possuem. O que é bastante benéfico, uma vez que esta analogia pode ser usada para prever os resultados da aplicação deste modelo.

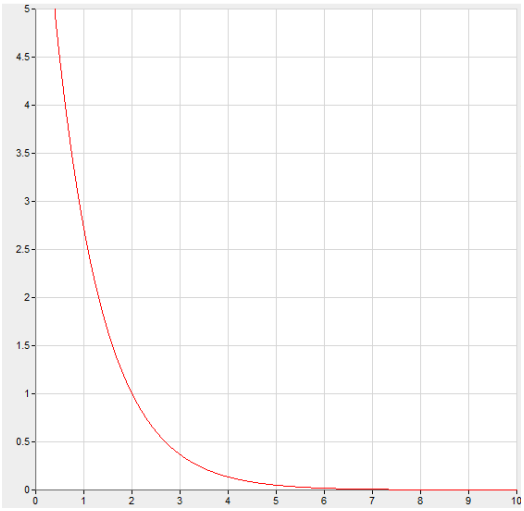

Figura 2.10 – Gráfico de uma exponencial inversa semelhante à decomposição dos químicos presentes no filme de uma máquina fotográfica. O eixo das abcissas representa a quantidade de luz recebida e o eixo das ordenadas o químico restante.

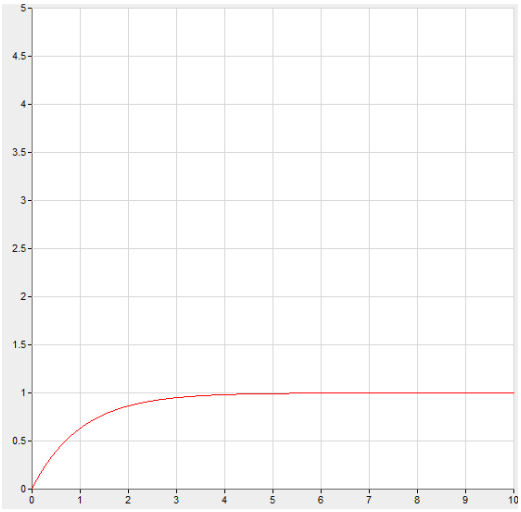

Figura 2.11 – Gráfico que representa o impacto da quantidade de luz recebida pelo filme na intensidade da luz na imagem final. O eixo das abcissas representa a quantidade de luz recebida e o eixo das ordenadas a intensidade da luz.

3 Trabalho Relacionado

A Radiosidade tem despertado muito interesse junto da comunidade de síntese de imagem. Em primeiro lugar porque o tema continua a ser actual, pois existem muitos problemas em aberto cuja solução não existe ou pode ser melhorada. Segundo, porque a demanda do mercado por ferramentas gráficas capazes de produzir imagens cada vez mais realistas, usando hardware comum sem sacrificar a capacidade de produção.

O realismo depende de modelos matemáticos para ser consistente, pelo que o facto da Radiosidade não se basear em modelos empíricos lhe confere grande valor na área.

Não existem dois sistemas de radiosidade iguais, todos eles têm características que os tornam únicos. O estudo aprofundado de cada um deles permite retirar lições importantes, para que os futuros sistemas sejam cada vez mais completos e para que não se caia em erros desnecessários.

3.1 RadiosGL[1]

Esta aplicação é muito representativa do que existe na área de Radiosidade. É composta por dois módulos, um efectua os cálculos de Radiosidade, o outro permite a visualização básica do ambiente virtual, à semelhança da solução proposta.

Apesar de ter nove anos de existência, implementa todos os algoritmos pilares da Radiosidade. Permite subdividir a geometria dinamicamente ou estaticamente com base num valor de referência. Usa o método do semicubo para calcular os factores de forma entre superfícies, permitindo mesmo mudar a sua dimensão.

Na Figura 3.1 é possível observar o resultado final, bastante característico, das primeiras implementações do Modelo de Iluminação Global por Radiosidade, onde as cores provenientes de fontes de luz não pontuais se misturam criando uma espécie de arco-íris.

[1] http://hcsoftware.sourceforge.net/RadiosGL/RadiosGL.html

Da mesma forma, é possível observar na Figura 3.2 as diferenças de intensidades de luz provenientes de uma fonte não pontual, onde o seu formato influencia visivelmente a distribuição da luz no ambiente virtual.

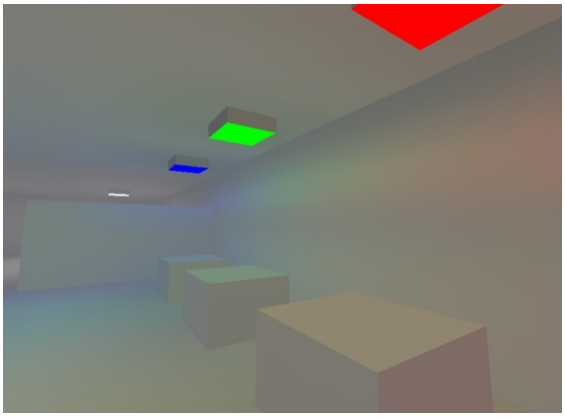

Figura 3.1 – Onde se pode observar a combinação de cores ao longo da parede.

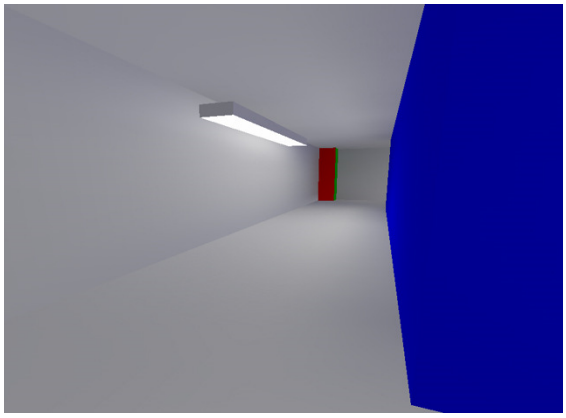

Figura 3.2 – Onde se pode observar as diferenças de intensidade ao longo da parede provocadas pela luz.

Esta aplicação usa o Algoritmo Progressivo de Radiosidade para propagar a radiação das superfícies pelo ambiente virtual.

Os autores desta aplicação não implementaram suporte para materiais com texturas, o que tem impacto nos ambientes virtuais onde grande parte da informação visual está presente nas texturas. Contrariamente o protótipo integra todas as texturas do ambiente virtual original, no ficheiro de resultados.

A aplicação suporta ficheiros de entrada do tipo MDL, e como ficheiro de saída usa um formato proprietário chamado *RGL scene file*. O uso de formatos proprietários pode ser problemático, uma vez que pode restringir a integração dos resultados com outras ferramentas. Para contrariar esta situação, o protótipo usa a especificação COLLADA [19] para importar informações sobre os ambientes virtuais e para registar resultados.

3.2 Radvis: Radiosity Visualisation[2]

Ao contrário de outras aplicações que usam modelo de Radiosidade, o objectivo principal do Radvis não é simplesmente aplicar este modelo, mas sim mostrar o seu funcionamento interno.

[2] http://www.cs.cmu.edu/~radiosity/radvis.html

O utilizador pode observar o funcionamento, passo a passo, da Solução Matricial de Radiosidade, do Algoritmo de Radiosidade Progressiva e da Solução Hierárquica de Radiosidade. Todos estes algoritmos são bastante representativos do estado da arte da área.

Ao dar a possibilidade ao utilizador de poder observar e parametrizar os diversos algoritmos, como se pode observar na Figura 3.3, esta aplicação dá um contributo académico bastante valioso. Permite a alunos da área da síntese de imagem (e áreas adjacentes) construir um modelo mental destes algoritmos de uma forma mais fácil que utilizando outros materiais didácticos.

Para além de ser possível observar o funcionamento dos algoritmos, é também possível traçar o gráfico da função de radiosidade de qualquer superfície do ambiente virtual. Desta forma é possível seguir a sua evolução ao longo do tempo e analisar alguns erros comuns que surgem num ambiente virtual mal preparado.

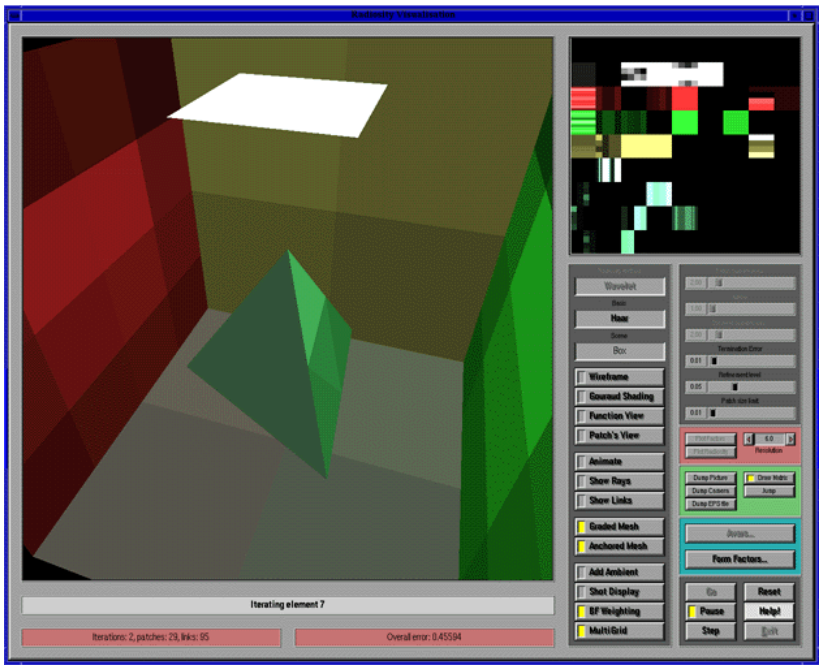

Figura 3.3 – O interface do Radvis permite ao utilizador parametrizar os diversos algoritmos de Radiosidade.

Apesar do protótipo não possuir esta componente didáctica, os resultados intermédios que este gera dão para analisar a evolução do Algoritmo de Radiosidade Progressiva.

3.3 RRV: Radiosity Renderer and Visualizer[3]

Esta aplicação, à semelhança do RadiosGL, é composta por dois módulos. O primeiro efectua os cálculos de intensidade das superfícies recorrendo ao Modelo de Iluminação Global por Radiosidade, o outro permite navegar no ambiente virtual em tempo-real.

Uma das grandes contribuições desta aplicação reside no facto de usar hardware para acelerar os cálculos de Radiosidade. Mesmo assim, se o ambiente virtual for demasiado complexo, o tempo de computação pode demorar dias.

O visualizador também permite ao utilizador observar o desenrolar das computações do Algoritmo de Radiosidade, à semelhança do Radvis. É possível visualizar o ambiente virtual com algumas técnicas de renderização desactivadas.

A Figura 3.4 mostra um dos modos de visualização que permite ver a geometria base do ambiente virtual. À semelhança da figura anterior, a Figura 3.5 mostra a geometria após a subdivisão.

Depois da radiosidade do ambiente virtual ter sido obtida para cada uma das superfícies, o utilizador pode observar os resultados sem interpolação (Figura 3.6) ou com interpolação (Figura 3.7). Com esta opção de visualização o utilizador pode analisar melhor o impacto que cada um dos algoritmos pré e pós-radiosidade tem na imagem final.

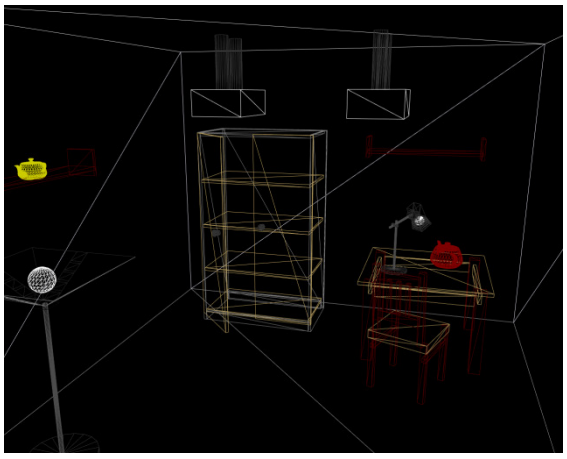

Figura 3.4 – Geometria inicial do ambiente virtual.

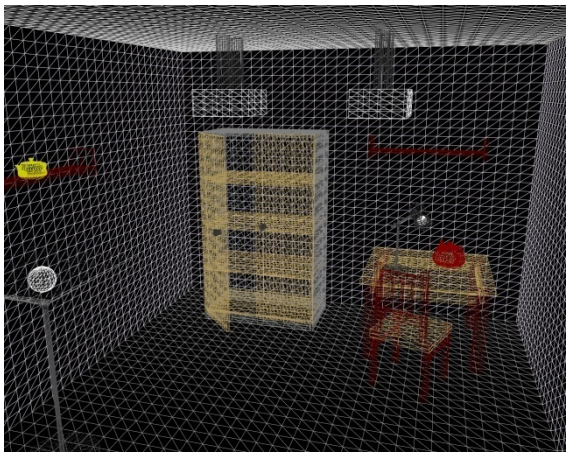

Figura 3.5 – Resultado da subdivisão de geometria para a aplicação do Algoritmo de Radiosidade.

[3] http://dudka.cz/rrv

Figura 3.6 – Resultado da computação do Algoritmo de Radiosidade sem interpolação.

Figura 3.7 – Resultado final, com interpolação nas superfícies.

Tal como a aplicação RadiosGL (Secção 3.1), esta também usa formatos proprietários de ficheiros, tanto de entrada como de saída. No entanto este formato está definido com base em XML e os autores disponibilizam o ficheiro DTD (*Document Type Definition*), o que facilita uma possível integração com outras aplicações.

As aplicações anteriormente analisadas fazem parte de estudos da área da Computação Gráfica que permitem uma melhor compreensão do Modelo de Iluminação por Radiosidade. Nem todas elas foram desenvolvidas com os mesmos objectivos em mente, mas todas elas deram o seu pequeno contributo para a área.

4 Arquitectura e Implementação

Esta secção tem como objectivo mostrar a arquitectura e detalhes da implementação do protótipo (RadioCOLLADA). Muitos destes detalhes estão relacionados com as tecnologias escolhidas e optimizações usadas.

Durante a implementação do protótipo muitas escolhas foram feitas, uma vez que existem muitas opções na forma como se implementar alguns subsistemas.

A evolução constante do hardware gráfico também contribui para este grande leque de opções, mas a escolha arbitrada do Algoritmo de Radiosidade Progressiva não favorece todas da mesma forma. Esta secção tem por objectivo mostrar as opções que foram tomadas, bem como abordar possíveis alternativas de implementação que não chegaram a ocorrer, ou que foram descartadas.

4.1 Módulo de Renderização de Radiosidade

O módulo de Renderização de Radiosidade é responsável por aplicar o Modelo de Iluminação Global por Radiosidade aos ambientes virtuais. Este processo pode ser moroso e depende da complexidade e número de elementos que o ambiente virtual possua.

Para obter os valores de radiosidade por elemento é necessário alocar estruturas de dados e preenchê-las com dados que são usados no decorrer do processo. Para além disso, é necessário reunir informação sobre o próprio ambiente virtual e as relações que os elementos têm entre si.

As etapas necessárias que o protótipo utiliza para obter os valores de radiosidade estão descritas na Figura 4.1. Cada um dos pontos presentes será analisado em profundidade nas próximas subsecções.

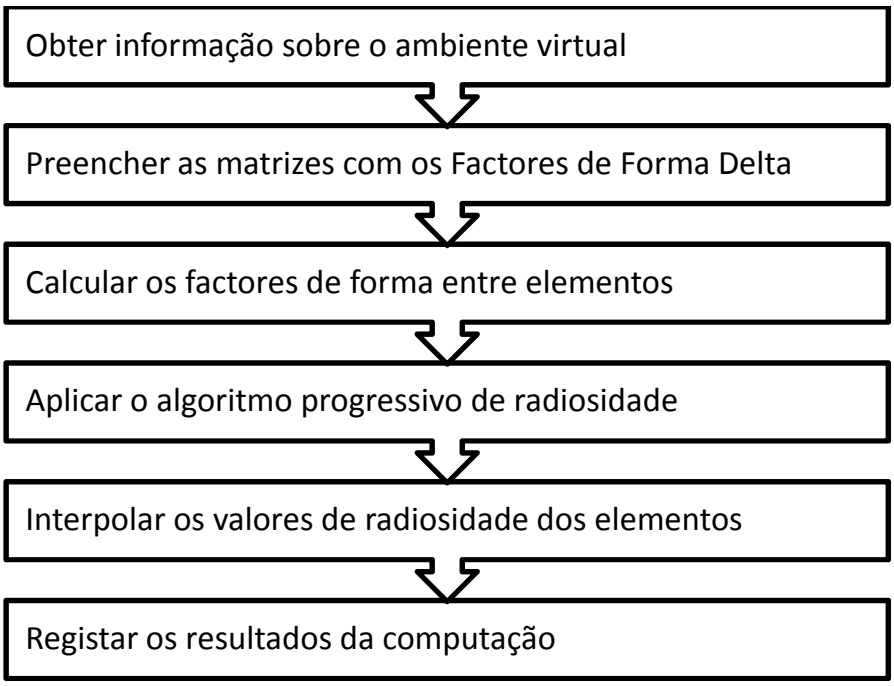

Figura 4.1 – Etapas que o protótipo percorre para aplicar o Modelo de Iluminação Global por Radiosidade.

Para aplicar o Modelo de Iluminação Global por Radiosidade utilizando o protótipo é necessário correr o módulo de renderização usando a seguinte sintaxe:

RENDERER -r inputFile outputFile [-hemiRes resolution] [-maxNumIterations numIterations] [-outputInterval numInterval] [-nearPlane nearValue] [-farPlane farValue]

Os parâmetros **inputFile** e **outputFile** são obrigatórios e representam o nome do ficheiro de entrada e saída, respectivamente. É possível alterar a resolução das faces do semicubo especificando o parâmetro **resolution**. A resolução por defeito é 512, que equivale a guardar as imagens em texturas de 512 por 512 *texels*. Também é possível especificar o número máximo de iterações do Algoritmo de Radiosidade Progressiva que o protótipo pode fazer usando o parâmetro **maxNumIterations**. No fim dessas iterações os resultados parciais tornam-se os definitivos.

O protótipo, à medida que processa iterações, vai emitindo ficheiros com os resultados parciais. Estes resultados são estruturalmente iguais ao ficheiro final e podem ser visualizados como tal, no entanto não têm os valores de radiosidade finais. O parâmetro **numInterval** especifica com que regularidade (número de iterações) estes ficheiros de resultados parciais são gerados.

Como à partida o protótipo desconhece a escala do ambiente virtual, é possível especificar, através dos parâmetros **nearValue** e **farValue**, a distância focal do plano de projecção e do plano de corte, respectivamente. Estes dois valores devem ser alterados com algum cuidado, caso contrário as imagens geradas para os semicubos podem não conter todos, ou mesmo nenhum dos elementos que nelas deveriam constar.

4.1.1 Obter informação sobre o ambiente virtual

Quando o protótipo é executado com o intuito de aplicar radiosidade a um ambiente virtual, o nome do ficheiro que contém informação sobre ambiente virtual é recebido por parâmetro.

Os dados presentes no ficheiro têm de estar estruturados consoante o esquema *COLLADA Digital Asset Exchange* [19]. A especificação do formato é aberta e a sua adopção não incorre em encargos monetários. É mantida pelo *Kronos Group*, sendo este responsável por manter as especificações de tecnologias como o OpenGL e OpenCL [14]. Para além disso, é baseada em XML, o que faz com que a sua integração e validação seja mais acessível. A maioria do software de modelação em 3D (ex: Blender v2.49, 3D Studio Max 2010, Maya 2009) é capaz de exportar e importar este formato.

Os ficheiros COLLADA são compostos por diversas secções (Figura 4.2), mas nem todas são analisadas na etapa de obtenção de informação sobre o ambiente virtual. O protótipo começa por triangular toda a geometria presente na secção **library_geometries** (Figura 4.3), caso esta não seja composta por triângulos, uma vez que a especificação do formato COLLADA suporta vários tipos de geometrias primitivas e o protótipo suporta apenas triângulos. Nesta secção existem vários tipos de listas (linha 4 à 34 da Figura 4.3) que armazenam dados como: pontos, normais e cores. Estes dados são indexados (linha 38 à 43), com o intuito de formar a armação do objecto em questão.

Depois da geometria estar toda sob a forma de triângulos, a secção **scene** (linha 37 à 38 da Figura 4.4) é analisada. Apenas a primeira **instance_visual_scene** é considerada pelo protótipo, uma vez que este não suporta mais que um ambiente virtual por ficheiro. Caso exista mais que um ambiente virtual no ficheiro, é emitido um alerta e só o primeiro ambiente virtual será considerado. Os nós que se encontram dentro do elemento **visual_scene** (linha 2 à 35 da Figura 4.4) são todos inspeccionados para achar aqueles que são objectos geométricos

no ambiente virtual. Os nós do elemento **visual_scene** também descrevem a posição dos objectos no ambiente virtual, bem como os materiais de que são compostos.

Os materiais presentes no ambiente virtual são descritos na secção **library_materials**. Estes descrevem várias propriedades que influenciam a aparência final dos objectos. Do ponto de vista do protótipo, as propriedades mais importantes são a componente da luz difusa e a componente emissiva de luz. A componente de luz difusa é usada para quantificar a radiosidade do material. Os objectos que tenham materiais com componente emissiva são as únicas fontes de luz do ambiente virtual. Como o Modelo de Iluminação Global por Radiosidade se baseia em leis da física para descrever o comportamento da luz, e na natureza não existem focos de luz pontual, a única forma de introduzir fontes de luz é usando superfícies emissoras de luz. Se o material tiver texturas associadas, estas são também incluídas no resultado final.

```
1    <?xml version="1.0" encoding="UTF-8"?>
2    <COLLADA xmlns="http://www.collada.org/2005/11/COLLADASchema"
     version="1.4.0">
3        <asset>
16       <library_cameras>
29       <library_effects>
101      <library_lights>
113      <library_materials>
121      <library_geometries>
211      <library_visual_scenes>
247      <scene>
250  </COLLADA>
```

Figura 4.2 – Excerto do ficheiro COLLADA onde estão presentes as principais secções.

A classe abstracta **Scene**, do protótipo, é responsável por definir a interface que permite aceder aos dados do ambiente virtual de forma transparente. Por sua vez, existe uma classe chamada **ColladaScene** que estende da classe **Scene** e implementa os mecanismos necessários para aceder à informação contida nos ficheiros COLLADA. Desta forma, nenhuma estrutura interna de dados do protótipo está dependente de nenhuma especificação externa.

O acesso aos dados nos ficheiros COLLADA faz-se através de um *middleware* chamado ColladaDOM, desenvolvido pela *Sony Computer Entertainment Inc*. Este *middleware* é responsável por garantir que os ficheiros COLLADA estão de acordo com as especificações. É também responsável por abstrair o acesso aos dados usando estruturas que representam os elementos descritos nos ficheiros, bem como a possibilidade de fazer pesquisas nesses ficheiros.

```
<library_geometries>
    <geometry id="ReceiverPlane-Geometry" name="ReceiverPlane-Geometry">
        <mesh>
            <source id="ReceiverPlane-Position">
                <float_array id="ReceiverPlane-Position-array" count="12">1 1 0 1 -1 0 -1 -1 0 -1 1 0</float_array>
                <technique_common>
            </source>
            <source id="ReceiverPlane-Normals">
                <float_array id="ReceiverPlane-Normals-array" count="3">0 -0 1</float_array>
                <technique_common>
            </source>
            <source id="ReceiverPlane-Colors">
                <float_array id="ReceiverPlane-Colors-array" count="24">0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1
                </float_array>
                <technique_common>
            </source>
            <vertices id="ReceiverPlane-Vertex">
                <input semantic="POSITION" source="#ReceiverPlane-Position"/>
            </vertices>
            <triangles count="2" material="ReceiverMaterial">
                <input offset="0" semantic="VERTEX" source="#ReceiverPlane-Vertex" set="0"/>
                <input offset="1" semantic="NORMAL" source="#ReceiverPlane-Normals" set="0"/>
                <input offset="2" semantic="COLOR" source="#ReceiverPlane-Colors" set="0"/>
                <p>0 0 0 3 0 1 2 0 2 2 0 3 1 0 4 0 0 5</p>
            </triangles>
        </mesh>
    </geometry>
    <geometry id="EmissionPlane-Geometry" name="EmissionPlane-Geometry">
</library_geometries>
```

Figura 4.3 – Exemplo de uma secção de um ficheiro COLLADA onde está descrita a geometria dos objectos do ambiente virtual.

```
<library_visual_scenes>
    <visual_scene id="Scene" name="Scene">
        <node id="ReceiverPlane" name="ReceiverPlane" layer="L1">
            <matrix>4.97072 0 0 0 0 4.97072 0 0 0 0 4.97072 -2 0 0 0 1</matrix>
            <instance_geometry url="#ReceiverPlane-Geometry">
        </node>
        <node id="EmissionPlane" name="EmissionPlane" layer="L1">
        <node id="Lamp" name="Lamp" layer="L1">
        <node id="Camera" name="Camera" layer="L1">
    </visual_scene>
</library_visual_scenes>
<scene>
    <instance_visual_scene url="#Scene"/>
</scene>
```

Figura 4.4 – Neste excerto de XML está descrito um ambiente virtual, bem como, as referências para os objectos que o compõem.

A relação entre a classe **Scene** e a classe **ColladaScene** está presente no diagrama da Figura 4.5. Para que o protótipo suportasse outro formato de ficheiro, bastaria criar uma classe que estendesse da classe **Scene**. Ao implementar os seus métodos abstractos, com os mecanismos necessários para ler, escrever e subdividir a geometria em triângulos, o protótipo estaria apto para lidar com o novo formato.

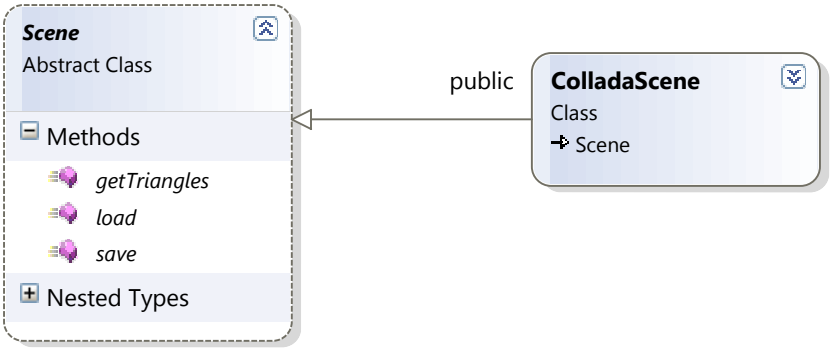

Figura 4.5 – Diagrama que mostra a relação entre a classe Scene e ColladaScene.

4.1.2 Preencher as matrizes dos Factores de Forma Delta

Para calcular os factores de forma entre elementos, o protótipo recorre a duas matrizes bidimensionais com o tamanho igual à resolução das faces do semicubo usado.

Figura 4.6 – Representação gráfica da matriz do topo do semicubo. Tons escuros equivalem a valores perto de 0, tons claros a valores perto de 1.

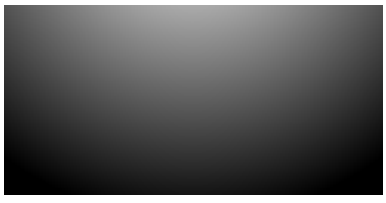

Figura 4.7 – Representação gráfica da matriz de uma das faces laterais do semicubo.

Por cada uma das células das matrizes o factor de forma é calculado como está descrito na Secção 2.5 e armazenado na própria célula. A distribuição destes valores da face do topo do semicubo é radial, como se pode ver na Figura 4.6, uma vez que é a 90º em relação à superfície que esta emite mais energia. Nas faces laterais a distribuição dos valores é diferente. À medida que o ângulo tende para zero em relação à superfície, os valores, também eles, tendem para zero, como se pode observar na Figura 4.7.

Para manter o equilíbrio energético nos ambientes virtuais, é necessário que a soma de todas as células das matrizes de todas as superfícies do semicubo seja inferior ou igual a 1. Caso contrário, as superfícies dos elementos poderiam emitir mais energia do que a que recebiam. Esta situação faria com que o ambiente virtual ganhasse energia em vez de a perder. O ambiente virtual perde energia no sentido em que parte dela é emitida para o vazio. No caso do Algoritmo de Radiosidade Progressiva, como este valor nunca tenderia para zero, no limite nunca iria existir uma situação de equilíbrio energético, portanto não existiria solução final.

Era possível armazenar estas matrizes em ficheiros para mais tarde serem consultadas. Desta forma, não seria necessário estar a proceder sua construção todas as vezes que se pretende aplicar radiosidade a um ambiente virtual. No entanto, como a resolução das faces do semicubo é um parâmetro do protótipo, tornaria muito dispendioso a armazenamento de todas as matrizes. Mesmo limitando o número de resoluções possíveis, o que faz sentido pois as placas gráficas têm limitações nas resoluções de textura que suportam, o cálculo destas matrizes é inferior a 1% do tempo total de processamento.

4.1.3 Calcular os factores de forma entre elementos

A obtenção dos factores de forma entre elementos é um dos passos mais morosos. Esta é umas das etapas que mais favorece com optimizações, uma vez que é uma das zonas críticas no desempenho do protótipo.

Nesta etapa, por cada elemento do ambiente virtual, todos os outros elementos são projectados no semicubo[4] deste. Esta projecção pode ser efectuada recorrendo a modelos analíticos ou recorrendo a hardware gráfico. O protótipo em questão usa hardware gráfico para acelerar esta etapa, conforme está descrito na Secção 2.9.1.

Para obter os factores de forma, são retiradas "fotografias" dos outros elementos do ambiente virtual usando as faces do semicubo como plano de projecção. As imagens obtidas

[4] É possível fazer projecções recorrendo a outros sólidos, como por exemplo Parabolóides (Secção 2.9.2).

não são mostradas no ecrã, mas sim renderizadas directamente para *buffers* internos da placa gráfica, deste modo com um melhor desempenho. Para conseguir renderizar o conteúdo do ambiente virtual para os *buffers* é necessário configurar o hardware gráfico para tal (neste caso através do OpenGL). Para o fazer é necessário criar um *Frame Buffer Object* (FBO), conforme se pode ver na segunda linha do excerto de código da Figura 4.8. O FBO por si só não faz nada, é necessário agregar outros objectos para que este ganhe funcionalidade. Como se pretende renderizar imagens, é necessário reservar memória no hardware gráfico para armazenar os píxeis resultantes. Criando uma textura (da linha 5 à 9 da Figura 4.8) com 32 bits de cor por píxel, tem-se um espaço de endereçamento de elementos considerável, uma vez que cada elemento vai ter uma cor distinta para que mais tarde seja possível proceder à sua identificação. No entanto, é necessário criar mais um objecto para que, nas imagens obtidas, a ordem dos elementos seja preservada. Esse objecto chama-se *Depth Buffer* e é criado e inicializado da linha 12 à 14 da Figura 4.8. Finalmente é necessário anexar estes dois objectos criados ao FBO, o que é feito da linha 17 à 19 da Figura 4.8. O diagrama da Figura 4.9 mostra o resultado final da configuração do FBO. Um pormenor importante é o facto da textura agregada ao FBO não possuir nenhum tipo de filtro, caso contrário a cor final dos píxeis poderia ser adulterada e deixaria de indexar o elemento certo.

Por cada elemento é obtido um conjunto de cinco imagens, uma para cada uma das faces do semicubo. Essas imagens têm um aspecto da Figura 4.10, onde a cor de um determinado píxel só repete se pertencer ao mesmo elemento. Analisando cada um dos píxeis destas imagens é possível estabelecer relações entre elementos que trocam energia entre si. No entanto, nem todos os píxeis têm o mesmo peso na quantidade de energia que transferem. Para quantificar essa transferência é necessário consultar as Matrizes dos Factores de Forma Delta calculadas na etapa anterior (Secção 4.1.2). Cada píxel da imagem tem uma coordenada correspondente na Matriz dos Factores de Forma Delta. O valor que estiver nesse elemento da matriz é o coeficiente de energia que esse píxel transmite a outro elemento. Como um elemento pode corresponder a vários píxeis na imagem, é necessário somar todos estes valores obtidos para saber o factor de forma entre os dois elementos.

```
1   //Generate framebuffer.
2   glGenFramebuffersEXT(1, &frameBuffer);
3
4   //Create texture to store the rendered image.
5   glGenTextures(1, &renderTexture);
6   glBindTexture(GL_TEXTURE_2D, renderTexture);
7   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
8   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
9   glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA8, hemiFaceSize,
    hemiFaceSize, 0, GL_RGB, GL_UNSIGNED_BYTE, NULL);
10
11  //Create depthbuffer, so that Z-Buffer algorithm can be applied.
12  glGenRenderbuffersEXT(1, &depthBuffer);
13  glBindRenderbufferEXT(GL_RENDERBUFFER_EXT, depthBuffer);
14  glRenderbufferStorageEXT(GL_RENDERBUFFER_EXT,
    GL_DEPTH_COMPONENT24, hemiFaceSize, hemiFaceSize);
15
16  //Bind the depthbuffer and texture to the framebuffer.
17  glBindFramebufferEXT(GL_FRAMEBUFFER_EXT, frameBuffer);
18  glFramebufferTexture2DEXT(GL_FRAMEBUFFER_EXT,
    GL_COLOR_ATTACHMENT0_EXT, GL_TEXTURE_2D, renderTexture, 0);
19  glFramebufferRenderbufferEXT(GL_FRAMEBUFFER_EXT,
    GL_DEPTH_ATTACHMENT_EXT, GL_RENDERBUFFER_EXT, depthBuffer);
```

Figura 4.8 – Excerto de código que redirecciona o destino das imagens renderizadas pelo hardware gráfico para um buffer na VRAM.

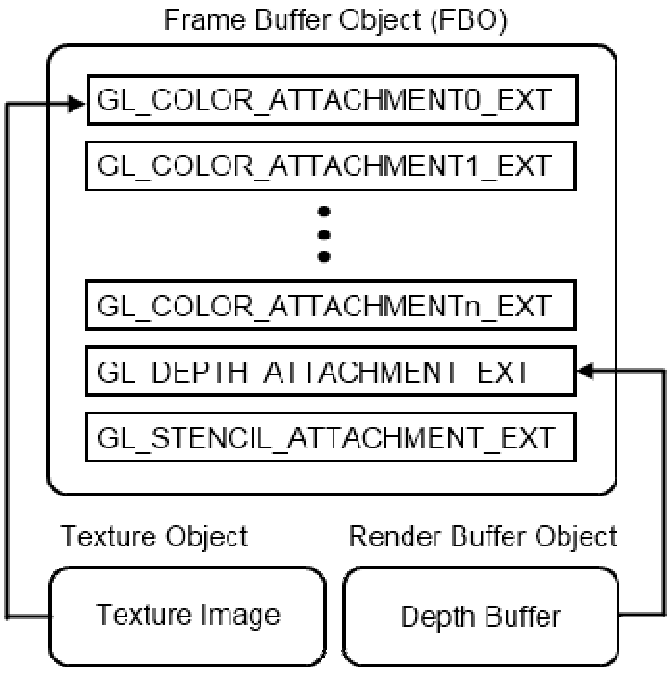

Figura 4.9 – Diagrama da configuração usada pelo protótipo para renderizar imagens *offscreen*.

É indispensável armazenar os factores de forma para que mais tarde o Algoritmo de Radiosidade Progressiva possa aceder a esta informação para poder calcular a transferência de energia entre elementos, conforme está descrito na Secção 2.4.

Inicialmente o protótipo usava uma estrutura de dados implementada na classe **PatchViewFactorCache**, presente no diagrama da Figura 4.11. No entanto, a operação de achar todos os elementos que trocam energia em relação a um determinado elemento apresentava uma complexidade linear O(n). Pois era necessário iterar sobre todos os pares de elementos para descobrir aqueles que continham o elemento. Só assim se sabia para quem enviar a energia do elemento, durante a computação. Este facto tornava o cálculo de radiosidade, em ambientes virtuais complexos, muito demorado. Para acelerar esta operação, cada elemento ficou responsável por armazenar a relação que tem com os outros elementos do ambiente virtual. Para o efeito, os elementos possuem uma lista interna onde registam os identificadores únicos dos elementos para os quais possuem um coeficiente de factor de forma superior a zero, e o próprio coeficiente de factor de forma entre os elementos. Quando chega a sua vez de emitir energia, percorrem a lista e usam os coeficientes para quantificar (com base na energia que possuem) a energia em enviam para esse elemento.

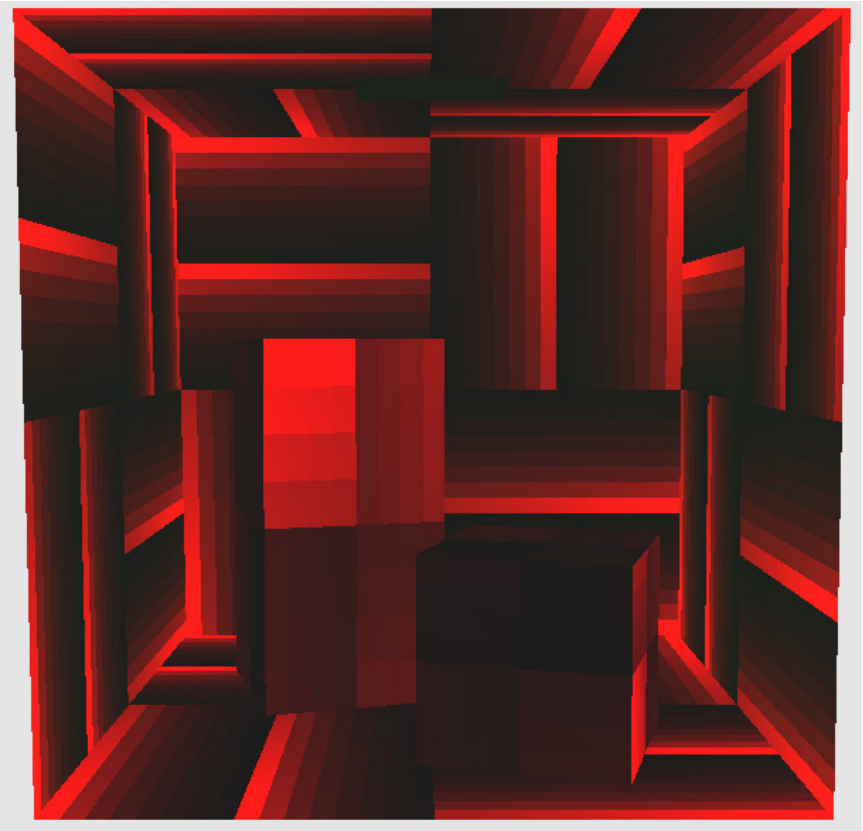

Figura 4.10 – Exemplo das Caixas Clássicas de Cornell, onde cada um dos elementos tem uma cor distinta. Esta cor serve de identificador único. Nenhum dos tons de vermelho na imagem é exactamente igual.

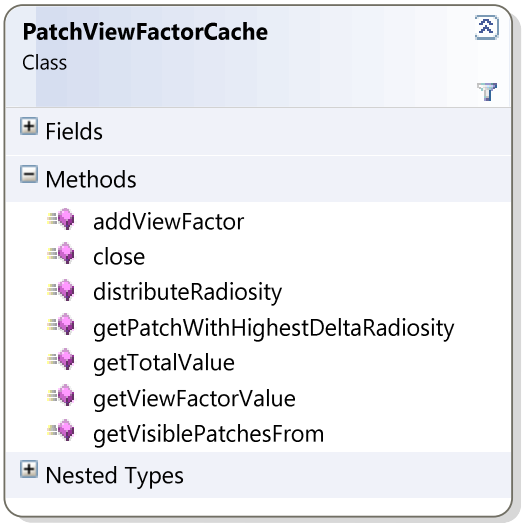

Figura 4.11 – Descrição da classe PatchViewFactorCache, usada para armazenar os factores de forma entre elementos.

4.1.4 Aplicar o Algoritmo de Radiosidade Progressiva

Para distribuir a energia do ambiente virtual entre os elementos, o protótipo usa o Algoritmo de Radiosidade Progressiva (Secção 2.7). Como o seu nome indica, este algoritmo é progressivo, ou seja, é aplicado continuamente ao ambiente virtual até à obtenção de uma solução aceitável ou a solução óptima.

Por cada iteração do algoritmo, o elemento com mais energia a emitir é seleccionado. Para o fazer, todos os elementos são ordenados com base no seu valor, usando a equação (4.1) para o obter, onde ΔB_j é a radiosidade por emitir do elemento j, $Area_j$ a área da superfície do elemento j e E_j a energia por emitir do elemento j.

$$E_j = \Delta B_j \times Area_j \qquad (4.1)$$

Após a obtenção do elemento com mais energia a emitir, é necessário proceder à sua distribuição. Nesta etapa, a informação sobre os factores de forma, obtidos na etapa anterior (Secção 4.1.3), é usada para quantificar a fracção de energia emitida para cada um dos outros elementos. No fim desta distribuição, o valor de radiosidade a emitir é colocado a zero, pois assume-se que o resto da energia se dissipou para o vazio.

O desempenho desta etapa dependente fortemente da velocidade do CPU do computador. Ao contrário da etapa anterior (Secção 4.1.3), esta não depende do hardware gráfico.

Com a massificassão de processadores com múltiplas *cores*, a utilização de APIs como OpenMP (*Open Multi-Processing*) [20] é cada vez mais vantajosa, não só em computadores de produção como em computadores pessoais. Ao processar mais elementos do que apenas o que tem maior energia por emitir, conforme está descrito no Algoritmo de Radiosidade Progressiva (Secção 2.7), é possível acelerar o processo do cálculo da radiosidade.

4.1.5 Interpolar os valores de radiosidade dos elementos

A interpolação dos valores de radiosidade é uma etapa muito importante para que os ambientes virtuais obtenham um nível de foto-realismo desejado. Antes da interpolação dos valores de radiosidade, os ambientes virtuais têm o aspecto presente na Figura 4.14 e Figura 4.15. Após a interpolação, obtêem o aspecto presente na Figura 4.16.

Na primeira etapa (Secção 4.1.1), o ambiente virtual é analisado. A partir daí, o resto das etapas ficam com acesso a um conjunto de elementos que compõem esse ambiente virtual, sem informação que identifique o objecto de onde os elementos são provenientes. Naturalmente, para aplicar interpolação entre elementos é necessário saber a sua proveniência, caso contrário, poderia ocorrer interpolação entre elementos de objectos distintos e produzir artefactos visuais que interfeririam com o foto-realismo do ambiente virtual.

Todos os elementos são obrigatoriamente criados por uma "fábrica", implementada na classe **ElementFactory**, presente no diagrama da Figura 4.12. Por cada objecto presente no ambiente virtual existe uma **ElementFactory** que cria todos os elementos provenientes desse objecto. Quando um elemento é criado, informações como vértices, normais e coordenadas de textura por vértice são indexadas e armazenadas na **ElementFactory** correspondente. Durante a interpolação dos valores de radiosidade, todos os índices dos vértices são percorridos e o método **getElementsWithVertexIndex** é invocado por cada um deles. Este método retorna os elementos que partilham um determinado índice. Ao fazer a média aritmética dos valores de radiosidade destes elementos, e colocando o resultado em cada um deles, os valores de radiosidade ficam interpolados.

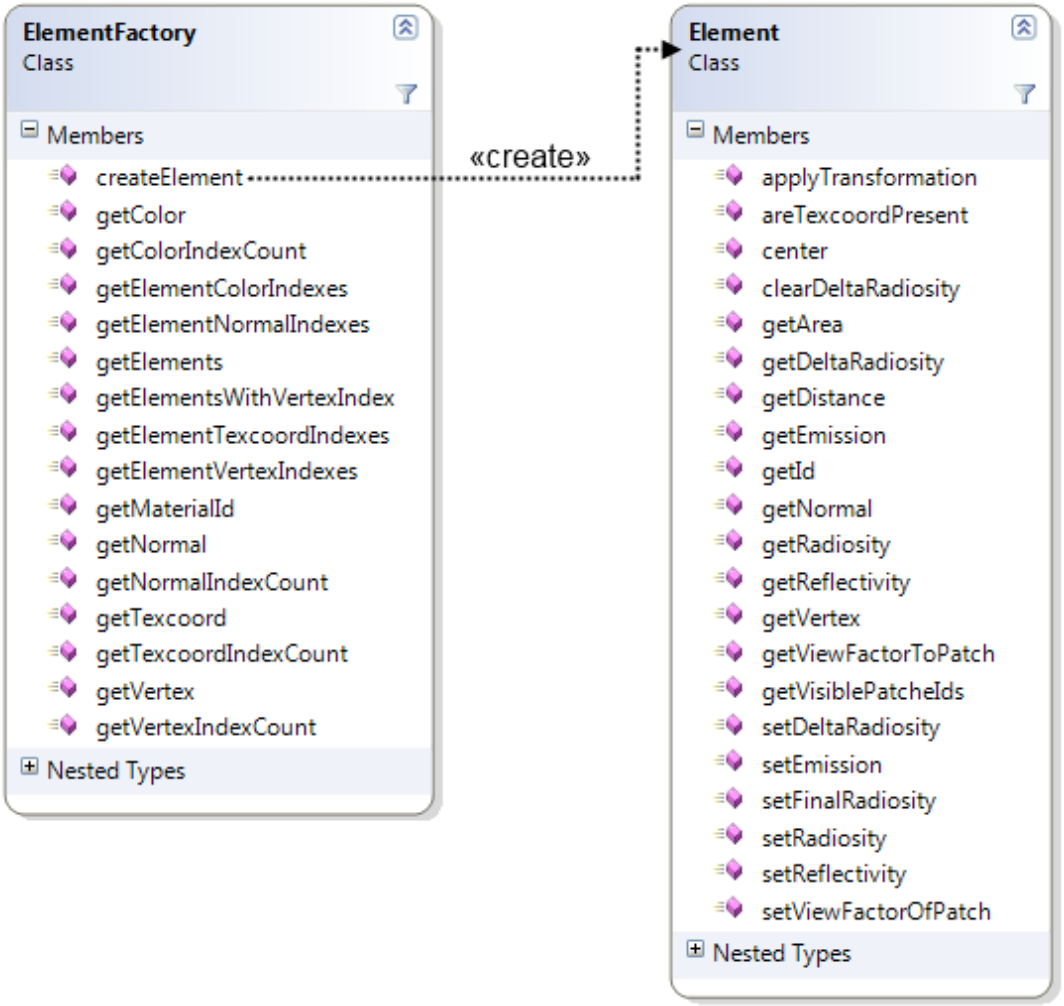

Figura 4.12 – Diagrama onde as classes que compõem relação Fábrica/Produto de elementos está representada, bem como os métodos públicos de cada uma delas.

4.1.6 Registar os resultados da computação

O registo dos resultados da computação é feita com base no ficheiro de entrada, pois o protótipo não suporta toda a especificação COLLADA, nem necessita, mas não pode omitir no ficheiro de resultados elementos que desconhece. Para contornar esta situação, o protótipo altera secções muito específicas para acomodar os resultados da computação. Começando pela secção **asset** (Figura 4.13) do ficheiro de saída, onde estão descritas as informações acerca da origem do ficheiro, o protótipo altera o texto do elemento **authoring_tool** (linha 4 da Figura 4.13) e coloca lá o seu nome e versão actual. De seguida, todos os elementos de geometria da secção **library_geometries** (Figura 4.3) são eliminados e rescritos novamente, reflectindo os resultados dos cálculos. Finalmente, as referências dos elementos de geometria são actualizadas para apontar para os novos elementos criados.

```
1  <asset>
2      <contributor>
3          <author>Illusoft Collada 1.4.0 plugin for Blender -
   http://colladablender.illusoft.com</author>
4          <authoring_tool>RadioCOLLADA v0.1 BETA</authoring_tool>
5          <comments/>
6          <copyright/>
7          <source_data>..\Scenes\scene1.dae</source_data>
8      </contributor>
9      <created>Sat Dec 05 23:40:19 2009</created>
10     <modified>Sat Dec 05 23:40:19 2009</modified>
11     <unit meter="0.01" name="centimeter"/>
12     <up_axis>Z_UP</up_axis>
13 </asset>
```

Figura 4.13 – Exemplo da secção de um ficheiro COLLADA onde estão descritas informações acerca da origem do ficheiro.

Depois de todas estas modificações em memória do ficheiro de entrada, este é gravado com o nome do ficheiro de saída. A escrita do ficheiro é feita pelo ColladaDOM, que se encarrega de verificar se todas as referências estão bem feitas, colocar alguns atributos obrigatórios com os valores por defeito e validar o XML produzido.

4.2 Módulo de Exposição

Quando o protótipo aplica o Modelo de Iluminação Global por Radiosidade ao ambiente virtual, é produzido um ficheiro COLLADA com os resultados da computação. No entanto este ficheiro não pode ser usado em nenhum visualizador, uma vez que os valores das cores não estão uniformizados. Para uniformizar os valores das cores presentes no ficheiro é necessário expor o ambiente virtual. Esta operação é muito semelhante ao que acontece na área da fotografia e está descrita na Secção 2.10.

No fim da exposição do ambiente virtual, é produzido um ficheiro COLLADA. Este ficheiro já tem os valores das cores uniformizados. Qualquer software que permita visionar ambientes virtuais descritos em ficheiros deste tipo não terá qualquer problema em fazê-lo, pois nenhuma liberdade foi tomada para registar os resultados da aplicação do Algoritmo de Radiosidade Progressiva. Todos os dados presentes no ficheiro de entrada são preservados no ficheiro de saída.

Para expor um ambiente virtual usando o protótipo, é necessário invocá-lo com a seguinte sintaxe:

RENDERER -e kValue inputFile outputFile

O parâmetro **kValue** é um número real, que controla o tempo de exposição do ambiente virtual como está descrito na Secção 2.10. Os parâmetros **inputFile** e **outputFile** são os nomes dos ficheiros dos ambientes virtuais de entrada e de saída, respectivamente.

4.3 Módulo de Visualização

Uma das vantagens de usar um formato aberto de ficheiro de resultados é a possibilidade de outras aplicações poderem ser integradas no fluxo de trabalho. Como foi referido anteriormente, as especificações dos ficheiros COLLADA não sofreram nenhum tipo de alteração para armazenar a informação resultante da aplicação do Algoritmo de Radiosidade Progressiva. Logo, qualquer aplicação que implemente a especificação COLLADA pode ser um potencial visualizador.

O ColladaDOM, *middleware* usado pelo protótipo para abstrair o acesso aos ficheiros COLLADA, traz um visualizador que usa a própria biblioteca e serve também de exemplo na sua utilização. No entanto, este visualizador não contempla toda a especificação, particularmente a cor por vértice. Foi necessário implementar partes da especificação COLLADA para se poder visionar os resultados do protótipo.

Para usar o visualizador é necessário invocá-lo com a seguinte sintaxe:

VIEWER inputFile

A navegação no ambiente virtual faz-se usando o teclado e o rato. Os controlos estão presentes na Tabela 4.1.

Acção	**Descrição**
Botão esquerdo do rato e arrastar	Rodar a câmara.
Botão direito do rato e arrastar	Mover a câmara.
Roda de deslocação	Aproximar/Afastar a câmara.
W	Mover negativamente a câmara no eixo dos XX.
S	Mover positivamente a câmara no eixo dos XX.
A	Mover positivamente a câmara no eixo dos YY.
D	Mover negativamente a câmara no eixo dos YY.
X	Mover negativamente a câmara no eixo dos ZZ.
SPACE	Mover positivamente a câmara no eixo dos ZZ.
M	Acelerar os movimentos da câmara.

N	Abrandar os movimentos da câmara.
Q	Ligar/Desligar o modo *wireframe*.
L	Ligar/Desligar luzes por *hardware*.

Tabela 4.1 – Lista de controlos para manipular o ambiente virtual no visualizador.

4.4 Problemas e Soluções

O ambiente virtual presente na Figura 4.14, Figura 4.15 e Figura 4.16, tem como objectivo testar a dispersão da luz numa superfície. É composto por um plano e uma única fonte de luz em forma de esfera. Os elementos são visíveis devido a falta de interpolação dos resultados obtidos.

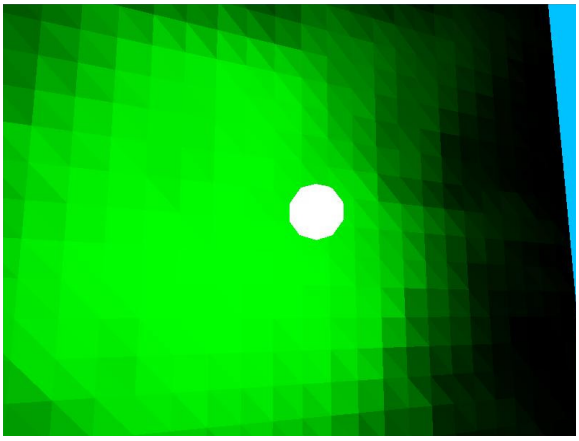

Figura 4.14 – Ambiente virtual que testa a dispersão da luz numa superfície sem interpolação entre elementos.

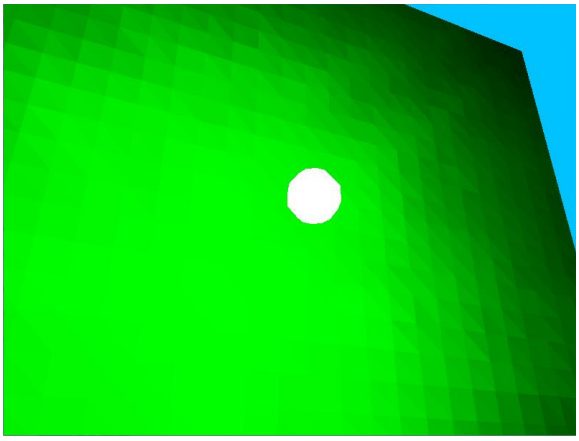

Figura 4.15 – Ambiente virtual que testa a dispersão da luz com alguns artefactos corrigidos.

Entre a imagem da Figura 4.14 e Figura 4.15, existe uma grande diferença na forma como a luz se espalha pela superfície. A falta de uniformização dos valores das matrizes dos Factores de Forma Delta é a causa dos artefactos visíveis na Figura 4.14. A soma de todos os Factores de Forma Delta tem de ser 1. Quando esta uniformização é aplicada, os artefactos desaparecem, como se pode observar na Figura 4.15.

A Figura 4.16 mostra o ambiente virtual anterior, renderizado pelo protótipo com interpolação entre elementos. É claramente visível a transição gradual da sombra para a luz, característica da utilização de uma exposição do ambiente virtual com um valor de K baixo. É muito importante escolher o valor de K com base nos resultados que se pretendem obter. Nem sempre o objectivo é realçar os detalhes nas zonas de sombra, por vezes os detalhes nas zonas

mais claras pode ser mais importantes. A imagem Figura 4.16 apresenta alguns artefactos provenientes do *aliasing* do semicubo. Para os eliminar, basta aplicar uma rotação no semicubo por cada elemento do ambiente virtual [9 pp. 86-87].

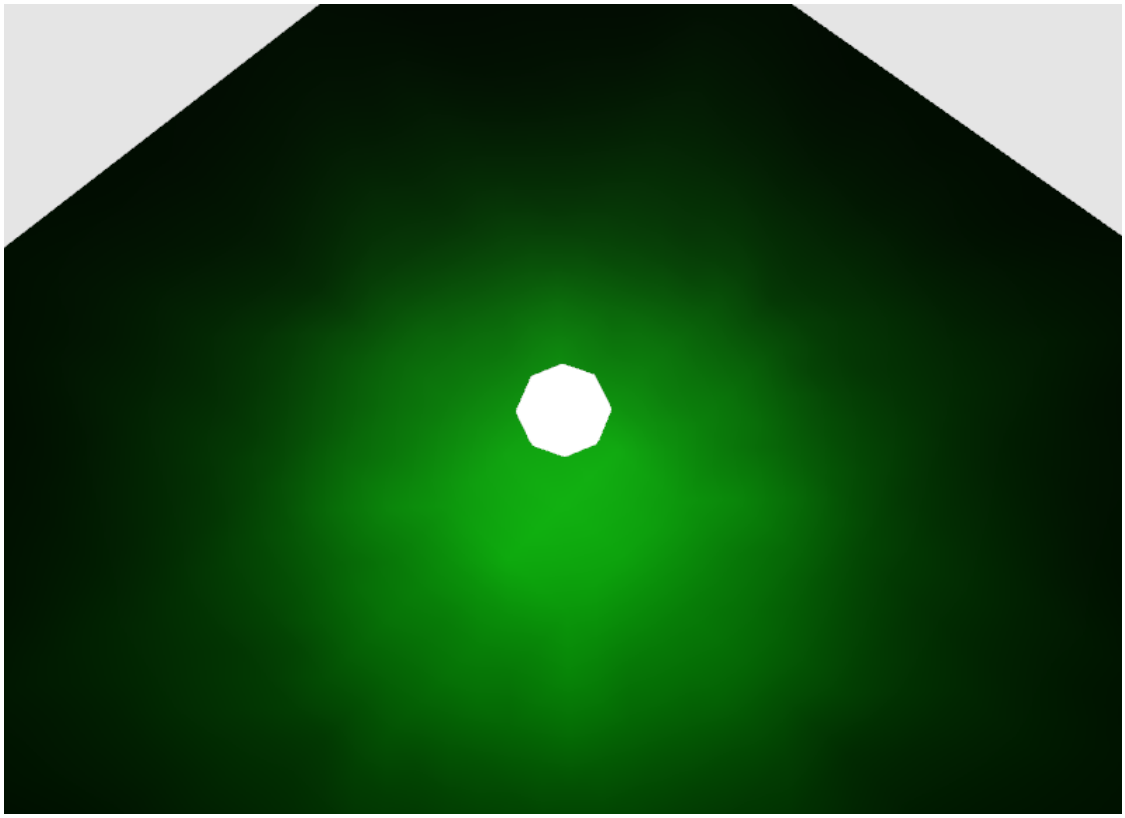

Figura 4.16 – Ambiente virtual que testa a dispersão da luz, com interpolação entre elementos.

O ambiente virtual da imagem da Figura 4.17 e Figura 4.18 é composto por duas paredes, uma vermelha e outra verde, onde o chão reflecte todo o tipo de luz e o tecto é uma superfície emissora de luz branca. O que se pretende testar com este ambiente virtual são os efeitos da luz indirecta na superfície do chão. Esta superfície apresenta a combinação das cores das paredes. Perto das arestas é possível ver, com mais intensidade, o contributo da luz reflectida das paredes mais próximas, que é uma das características dos modelos globais de iluminação por radiosidade.

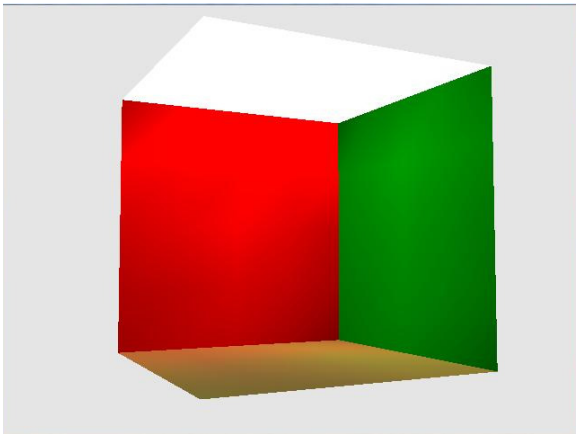

Figura 4.17 – Ambiente virtual que testa a transferência de cor entre superfícies.

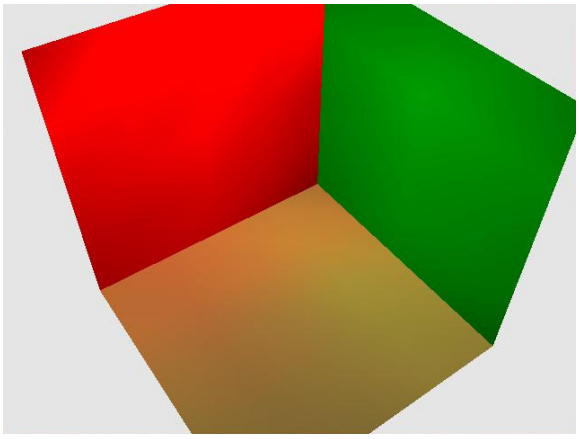

Figura 4.18 – Influência que as paredes coloridas têm na superfície do chão.

A imagem da Figura 4.19 mostra alguma incoerência no comportamento da luz. A luz verde atravessa a parede branca, que ficou amarela, e influencia o lado direito do ambiente virtual, o que não deveria acontecer. Esta situação ocorre quando as imagens do semicubo são renderizadas sem *Z-Buffer*. A falta de um *depthbuffer* associado ao FBO (ver Secção 4.1.3) é normalmente a causa desta situação. O mesmo ambiente virtual, mas sem este problema, está presente na imagem da Figura 4.20, onde a luz não invade o lado oposto em nenhum dos sentidos.

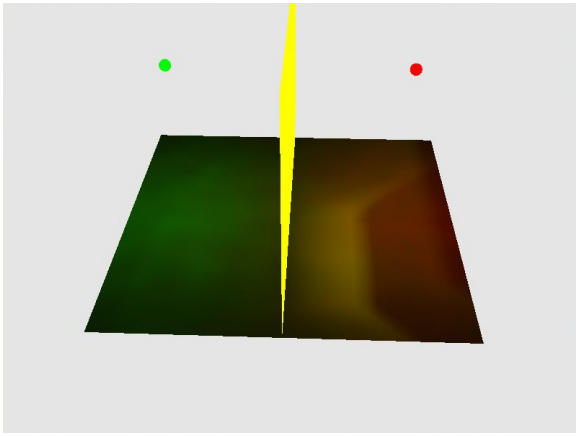

Figura 4.19 – Ambiente virtual onde está presente um erro na distribuição da luz.

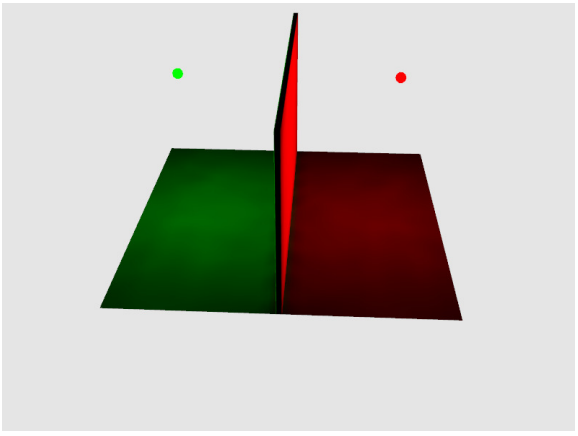

Figura 4.20 – Ambiente virtual sem problemas de *Z-Buffer*.

5 Testes e Resultados

Esta secção tem como objectivo descrever e analisar os testes realizados e os resultados obtidos. Os testes são provenientes do protótipo desenvolvido no âmbito desta dissertação. Foram criados com o intuito de mostrar o impacto que o Modelo de Iluminação Global por Radiosidade tem nos ambientes virtuais.

Os ambientes virtuais presentes na Figura 5.1 não apresentam, por si só, características foto-realistas. A utilização de ambientes virtuais simples permite observar melhor o comportamento da luz, e não provoca distracções provocadas por geometria complexa ou texturas com muito detalhe.

Foram escolhidos três ambientes virtuais para serem analisados:

- **"Parede":** Este ambiente virtual é constituído por uma parede branca, um plano de chão branco e uma fonte de luz branca direccionada para a parede. O que se pretende observar é o comportamento da luz em superfícies planas e na zona de intersecção da parede com o plano do chão. Este ambiente virtual é constituído por 4098 elementos (Figura 5.1-a).

- **"Xadrez":** Este ambiente virtual é constituído por uma parede verde e um chão com uma textura em xadrez. A parede não tem espessura e a sua largura é inferior ao do plano do chão. Com este ambiente virtual pretende-se analisar o comportamento da sombra provocada pela parede, a reflexão da luz proveniente da parede e a interacção da luz com a textura do chão. Este ambiente virtual é constituído por 2130 elementos (Figura 5.1-b).

- **"Caixas de Cornell":** Este é o ambiente clássico dos exemplos de radiosidade. É composto por uma espécie de sala com uma das paredes pintada de vermelho e a outra de verde. Tem uma única fonte de iluminação no tecto e no seu interior encontra-se um prisma quadrangular e um cubo. Todas as cores das superfícies, dimensões e posicionamento estão conforme o ambiente virtual original [5]. Este ambiente virtual é composto por 3842 elementos (Figura 5.1-c).

a)

b)

c)

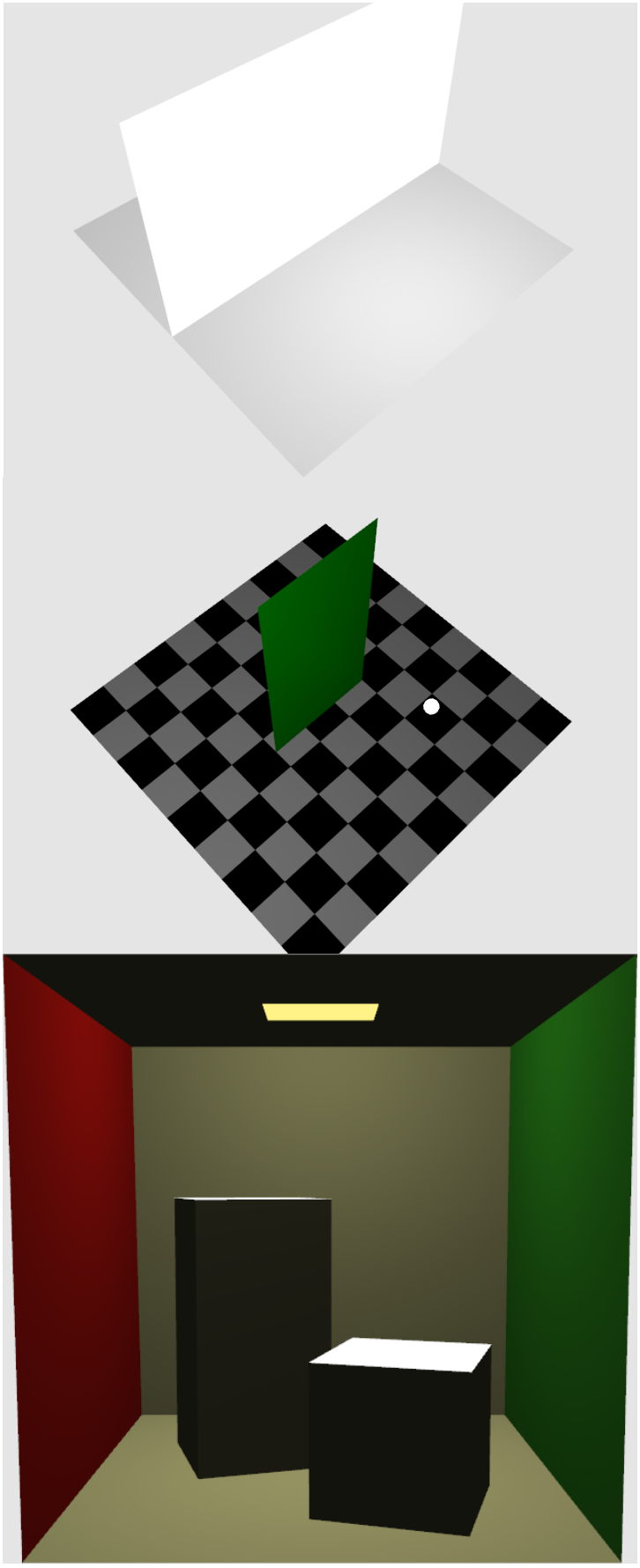

Figura 5.1 – Ambientes virtuais com iluminação por *hardware*. a) Parede. b) Xadrez. c) Caixas de Cornell.

5.1 Qualidade

Na sua essência, o Modelo de Iluminação Global por Radiosidade pretende aplicar as leis da física ao comportamento simulado da luz no ambiente virtual. A melhor forma de avaliar a qualidade da implementação do modelo é construindo maquetas reais dos ambientes virtuais, e depois comparar os resultados obtidos com as fotografias dessas maquetas. Este processo é muito moroso e envolve equipamento especializado para calibrar o ambiente virtual. Outra forma mais simples de avaliar os resultados é empiricamente, observando os ambientes virtuais em busca de potenciais discrepâncias com o comportamento percepcionado da luz.

A Figura 5.2-a corresponde ao ambiente virtual "Parede" (Figura 5.1-a) renderizado usando radiosidade. A dispersão da luz pelas superfícies passou de completamente uniforme, a uma distribuição mais focada nas superfícies. A fonte de luz (que não é visível nas imagens) está mais orientada para a parede. No entanto, o chão recebe luz indirecta da parede, como era previsível. A aresta que divide a parede do chão é perceptível mesmo com uma fonte de luz intensa projectada contra a parede. De facto, se tivermos em conta as Matrizes dos Factores de Forma (Secção 4.1.2), reparamos que para ângulos perto de 180º em relação à superfície os coeficientes tendem a ter valores muito próximos de zero (Figura 4.7), o que impede a transferência de energia entre elementos nestas condições, provocando uma zona mais escura ao longo da aresta.

Apesar do Modelo de Iluminação Global por Radiosidade descrever, do ponto de vista teórico, o comportamento da luz de forma muito realista, as suas implementações nem sempre o conseguem fazer da mesma forma. Particularmente aquelas, como este protótipo, baseadas na família de técnicas de amostragem por hemisfério. Na Figura 5.2-b é possível observar a sombra da parede verde. Esta sombra apresenta um contorno bem definido devido à elevada concentração de elementos no plano do chão. Quando não é o caso, é necessário aplicar técnicas de subdivisão de elementos (Secção 2.8). Para além da sombra, é possível ver a reflexão, no chão, da luz indirecta proveniente da parede verde. No entanto, a reflexão não se dá nas zonas pretas do padrão xadrez que o chão apresenta, como é espectável.

O ambiente da Figura 5.2-c é um exemplo clássico de um ambiente virtual de teste ao Modelo de Iluminação Global por Radiosidade. Nele é possível observar com clareza o esbatimento das cores entre superfícies e a própria cor da sombra, que deriva da cor da parede de onde proveio a luz reflectida indirectamente.

a)

b)

c)

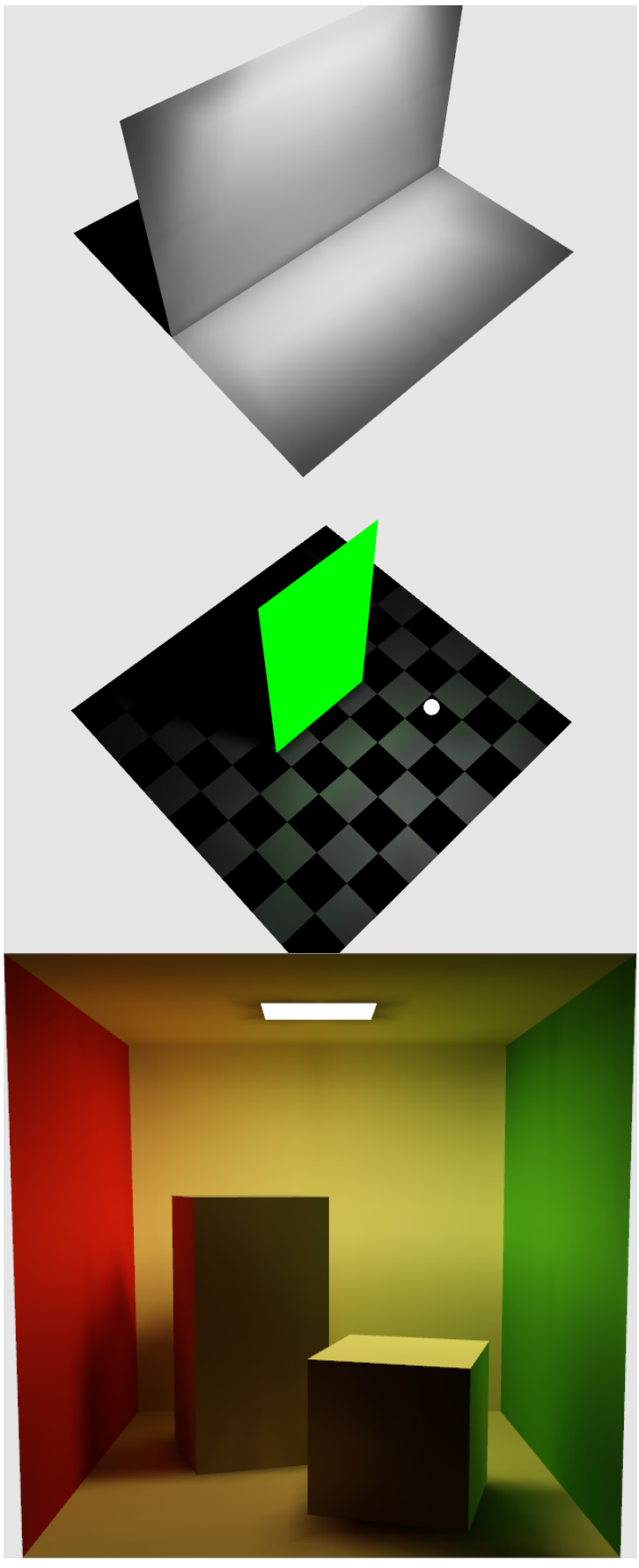

Figura 5.2 – Ambientes virtuais com iluminação por radiosidade. a) Parede. b) Xadrez. c) Caixas de Cornell.

Ao analisar com detalhe a Figura 5.2-c, são notórios artefactos visuais, especialmente à volta da fonte de luz. Estes artefactos surgem devido a uma subdivisão homogénea do ambiente virtual. A solução passaria por aplicar técnicas de subdivisão, *a priori* ou *a posteriori*, mais avançadas como *Adaptive Subdivision* [9] ou *Dynamic Subdivision* [6]. No entanto, do ponto de vista da radiosidade, os elementos estão a transferir energia entre si (na forma de luz) realisticamente. Se observarmos a sombra esverdeada do cubo, podemos concluir que, no mínimo, a luz que deu origem a essa tonalidade teve de interagir com a parede e depois com a face do cubo, antes de ser projectada no chão. São fenómenos como este que caracterizam as propriedades foto-realistas do Modelo de Iluminação Global por Radiosidade.

5.2 Desempenho

Tradicionalmente o tempo de computação necessário para aplicar o Modelo de Iluminação Global por Radiosidade a um ambiente virtual é bastante moroso. O processo pode ser acelerado usando hardware, como está descrito na Secção 2.9, ou optimizando as estruturas de dados que armazenam resultados intermédios. No protótipo implementado, foi usado hardware gráfico para optimizar a obtenção dos factores de forma, e como trabalho futuro, a paralelização das etapas do Algoritmo de Radiosidade Progressiva.

Os resultados foram obtidos com recurso a um computador com processador Intel Core 2 a 2.4GHz, 4GB RAM e uma placa gráfica NVIDIA GeForce GTX 260. O sistema operativo usado foi o Windows Vista 32bits com o Service Pack 2.

Foram testadas duas estruturas de dados em memória. A primeira consiste num mapa onde o par chave-valor é armazenado. A chave deste mapa corresponde à junção do identificador de dois elementos, em que as chaves [A, B] e [B, A] são consideradas a mesma, e o valor é o factor de forma entre o elemento A e o B. Os resultados da aplicação do algoritmo usando esta estrutura de dados estão presentes na Tabela 5.1.

A segunda estrutura de dados testada armazena os factores de forma numa lista individual que cada elemento possui. Cada item da lista é composto por um identificador de elemento e um coeficiente de factor de forma. Os resultados obtidos com esta estruturação estão presentes na Tabela 5.2.

Estruturação em Mapa			
Ambiente virtual	N.º de elementos	Tempo [minutos]	Memória [MB]
Parede	4098	157.1(3)	137
Xadrez	2130	4.2(6)	24
Caixas de Cornell	3842	264.88(3)	198

Tabela 5.1 – Resultados obtidos com o armazenamento dos factores usando uma estrutura do tipo mapa.

Estruturação em Lista			
Ambiente virtual	N.º de elementos	Tempo [minutos]	Memória [MB]
Parede	4098	3.7	216
Xadrez	2130	1.18(3)	28
Caixas de Cornell	3842	7.4(3)	325

Tabela 5.2 – Resultados obtidos com o armazenamento dos factores de forma directamente nos elementos, usando uma estrutura do tipo lista.

Os tempos de computação do protótipo com estruturação em mapa são muito superiores em relação aos de estruturação em lista, como se pode ver pelo gráfico da Figura 5.3. A causa é o custo linear da obtenção das chaves que contêm o identificador de um determinado elemento. Como na estruturação em lista os elementos registam individualmente com quem trocam energia, isto faz com que o custo da obtenção dos pares de elementos que trocam energia entre si seja nulo. Apesar da estruturação por lista apresentar um despenho superior, os custos de memória são mais elevados (Figura 5.4). Isto ocorre devido a redundância de informação que o armazenamento por lista tem. Como cada elemento guarda todos os outros elementos cujo factor de forma entre os dois seja superior a zero, na pior das hipóteses os valores dos factores de forma são duplicados. No gráfico da Figura 5.4 não é possível concluir que o consumo de memória seja exactamente o dobro, mas há que ter em conta a memória extra que a *C++ Standard Template Library* aloca para armazenar futuros dados.

Alocar os coeficientes dos factores de forma em memória (RAM ou VRAM) tem a vantagem do acesso ser muito mais rápido, pois os tempos de latência para estes componentes são muito mais baixos. Porém, quando o ambiente virtual é complexo e não existe memória suficiente para armazenar todos os coeficientes, é necessário arranjar estratégias alternativas. Uma possível alternativa passa por armazenar os valores em ficheiros. Deste modo, o consumo de memória deixaria de ser uma preocupação, mas os tempos de acesso à informação iriam ser superiores. Uma solução mais elegante seria agrupar os elementos e

renderizar o ambiente virtual faseadamente, submetendo os resultados directamente nas texturas dos objectos [21].

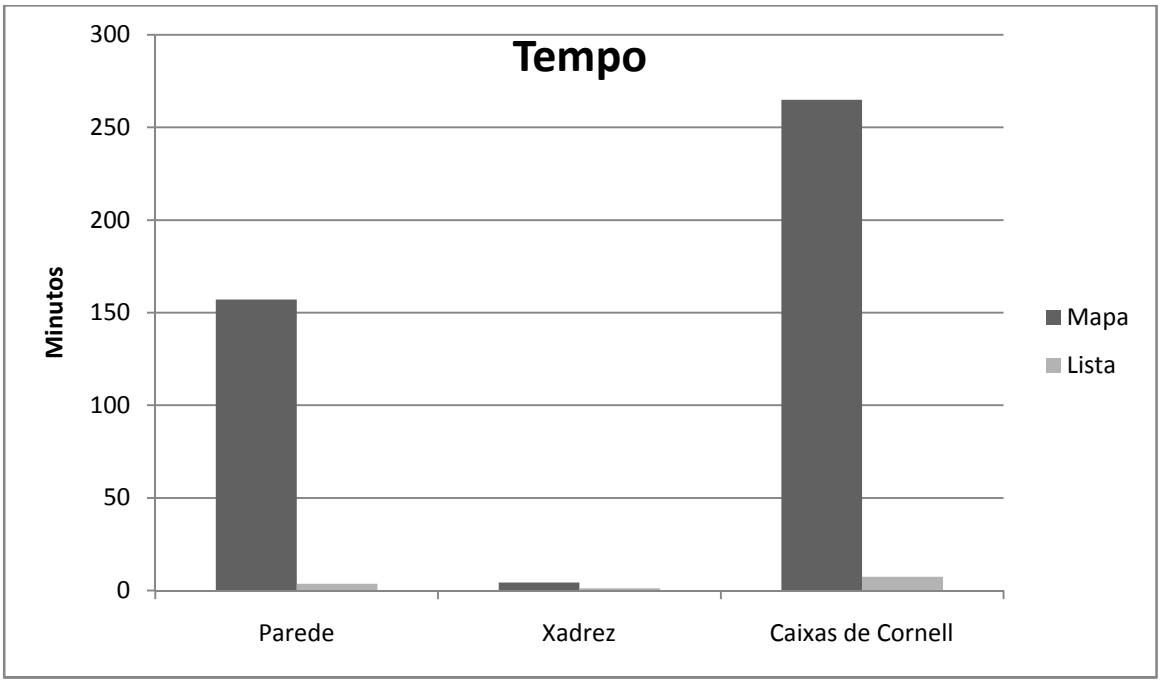

Figura 5.3 – Comparação dos tempos de renderização entre as duas estruturas de dados.

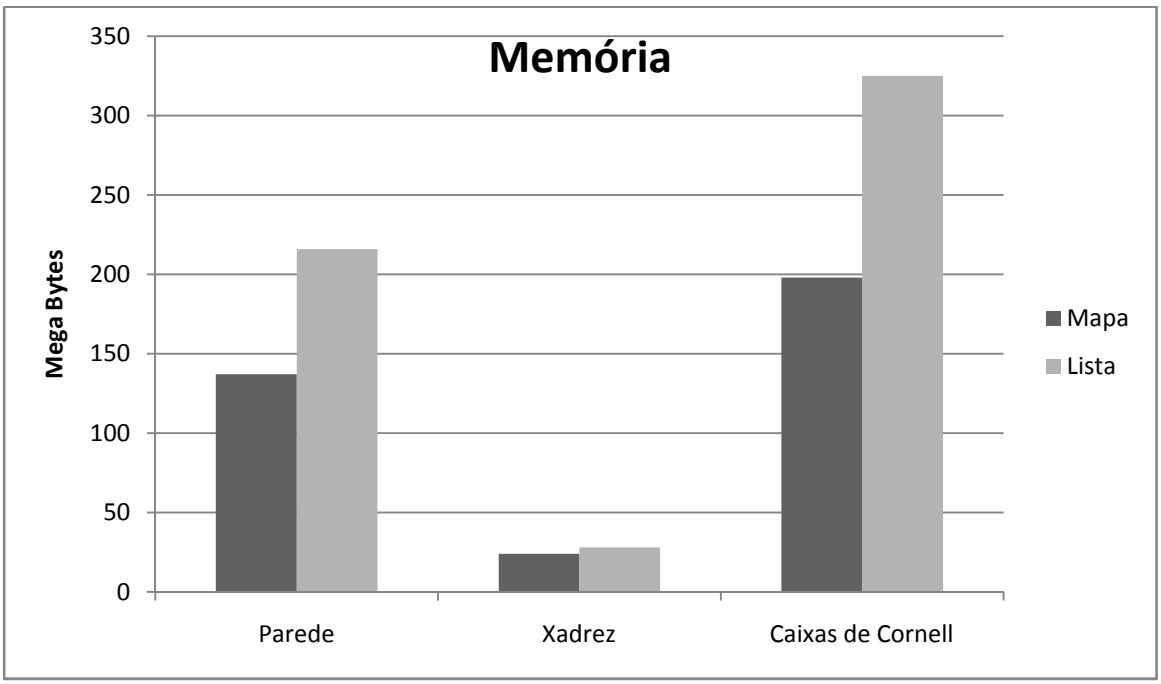

Figura 5.4 – Comparação da memória consumida durante a renderização, usando as duas estruturas de dados.

6 Conclusão

Ao longo desta Dissertação, o Modelo de Iluminação Global por Radiosidade foi analisado e o estado da sua arte foi levantado. Para aprofundar o conhecimento da área e testar hipóteses, foi desenvolvido um protótipo capaz de renderizar ambientes virtuais utilizando este modelo. Porém, o protótipo apresenta um consumo elevado de memória que o impede de ser aplicado a ambientes virtuais muito complexos.

Com a proliferação de hardware gráfico cada vez mais potente e de utilização mais genérica, é espectável que este possa trazer novas formas de aplicar radiosidade a ambientes virtuais e, com isso, melhor desempenho. Apesar do protótipo se basear nos pilares base do Algoritmo de Radiosidade Progressiva, como estão descritos por Cohen *et al*. [9], a utilização de hardware gráfico veio simplificar e acelerar algumas etapas do algoritmo, particularmente a obtenção dos factores de forma.

Inicialmente o protótipo apenas importava e exportava ambientes virtuais num formato próprio, à semelhança das aplicações presentes no Trabalho Relacionado (Secção 3). O uso de formatos próprios dificulta a utilização dos resultados produzidos. Mesmo assim muitos autores insistem em defini-los. Depois de uma análise às especificações do formato COLLADA [19], foi possível modificar o protótipo para que este passasse a usar estas especificações para obter informação dos ambientes virtuais e armazenar os valores de radiosidade. No entanto, nem todos os visualizadores de ambientes virtuais que suportam a especificação COLLADA estão preparados para lidar com valores provenientes dos cálculos de radiosidade, uma vez que estes não pertencem a nenhum conjunto fechado de valores e não podem ser utilizados directamente pelo hardware gráfico. Para garantir que qualquer visualizador, nas condições referidas anteriormente, possa renderizar o ambiente virtual, foi necessário introduzir uma segunda fase no processo de obtenção de ambientes virtuais com iluminação global por radiosidade. Essa fase consiste em submeter o ambiente virtual a um processo de exposição, à semelhança do que acontece na área da fotografia. Depois desta fase, obtem-se um ambiente virtual com a iluminação final capaz de ser processadas por qualquer visualizador que suporte a especificação COLLADA.

6.1 Trabalho Futuro

Ao longo desta dissertação foram identificadas temas relacionados que potencialmente têm um grande impacto na qualidade dos ambientes virtuais obtidos. No entanto, por se desviarem dos objectivos da dissertação, não foram desenvolvidos. Nesta secção pretende-se referir alguns desses temas.

6.1.1 Profundidade de campo na visualização dos ambientes virtuais

O Modelo de Iluminação Global por Radiosidade dá aos ambientes virtuais características foto-realistas no que diz respeito ao comportamento da luz. No entanto, se simplesmente os renderizarmos, continuam com características artificiais que degradam o nível de realismo. Uma dos aspectos que tem impacto, é o facto de não existir noção de profundidade, isto é, o sistema visual humano ajusta-se para observar a diferentes distâncias. Num ambiente renderizado de forma tradicional, sem profundidade de campo, não existe focagem.

Ao incluir o efeito de profundidade de campo num ambiente virtual [22] [23], para além de aumentar o realismo do mesmo, é possível direccionar a atenção do observador para objectos ou estruturas do ambiente virtual.

Num sistema de radiosidade em tempo-real, com profundidade de campo, o facto de partes do ambiente virtual ficarem desfocadas, pode ajudar na optimização do sistema, uma vez que não é necessário despender tantos recursos computacionais em zonas que vão ficar desfocadas. Num sistema de radiosidade em tempo-real, esta informação pode ser usada para alocar recursos computacionais a zonas do ambiente virtual que estejam a ser alvo de uma observação mais detalhada pelo observador.

6.1.2 Uso de BSP no acesso e armazenamento dos factores de forma

A implementação actual do protótipo favorece a velocidade em detrimento da memória utilizada. Inicialmente, com a utilização da estrutura de dados representada pela classe **PatchViewFactorCache**, o consumo de memória era inferior, mas os tempos de acesso eram muito elevados. Isto acontecia devido a necessidade de iterar por todo mapa das relações entre elementos, para obter todas as relações que envolviam um determinado elemento.

O uso de uma árvore gerada por BSP (*Binary Space Partitioning*), no auxílio do processo de obtenção das relações entre elementos, poderá fazer com que conjuntos significativos de

elementos sejam descartados por se encontrarem em áreas do ambiente virtual onde o elemento em estudo não tem visibilidade. Desta forma, o tempo de acesso aos factores de forma seria reduzido e estes poderiam estar armazenados em mapas, à semelhança do que acontecia na classe **PatchViewFactorCache.**

6.1.3 Obtenção dos Factores de Forma por Parabolóide

A obtenção dos factores de forma é um processo moroso por natureza, pois envolve quantificar a relação geométrica entre elementos do ambiente virtual, e estes normalmente estão presentes em grande número. A generalização do uso do hardware gráfico permitiu acelerar este processo. No entanto, o uso do semicubo na obtenção dos factores de forma ainda implica renderizar o ambiente virtual cinco vezes por amostra (elemento), uma para cada face do semicubo.

O uso de projecção para superfícies parabolóides é uma técnica que permite gerar texturas com reflexões em tempo-real [15]. Esta técnica permite reduzir de seis projecções, quando o mapeamento por cubo é usado, para apenas duas projecções, quando usados dois parabolóides para cobrir um campo de visão de 360º.

Para fazer uma amostragem dos factores de forma, bastaria apenas uma projecção para uma parabolóide colocado no ponto de amostragem. Como esta técnica, à semelhança da projecção para o semicubo, pode ser acelerada recorrendo a hardware gráfico, seria interessante avaliar os impactos no desempenho do cálculo dos factores de forma.

6.1.4 Implementação total em GPU (Graphics Processing Unit)

Conforme foi abordado muito superficialmente na Secção 2.9, uma das soluções consistia em colocar todo o trabalho do cálculo dos valores radiosidade no GPU.

Já existem soluções que aplicam radiosidade em tempo-real, sem nenhum tipo de atalho que sacrifique a qualidade do ambiente virtual. Um exemplo disso está presente em *GPU gems 2* [24]. No entanto, este método usa *shaders* [8] para optimizar o desempenho, e este ronda os 2fps no ambiente virtual das Caixas de Cornell, composto por 10000 elementos.

Uma das vantagens de uma implementação total em GPU seria um melhor desempenho na obtenção dos factores de forma. Como o acesso do GPU à VRAM é directo, já não seria necessário copiar o conteúdo dos buffers das imagens da VRAM para a RAM. Outra grande vantagem poderia ser a renderização em tempo-real de ambientes virtuais usando o Modelo de

Iluminação Global por Radiosidade. As especificações do OpenCL [14] permitem a utilização, em simultâneo, do OpenCL e do OpenGL. Existem formas de trocar informação entre estas duas frameworks. Desta forma, o OpenCL ficava responsável pelos cálculos e o OpenGL pela sua apresentação.

Implementar o Modelo de Iluminação Global por Radiosidade numa arquitectura como OpenCL apresenta desafios interessantes. Esta arquitectura é optimizada para *stream processing*, o que implica alterações profundas nas estruturas de dados. Neste tipo de arquitecturas, o processamento faz-se através de *kernels* que têm acesso a vários tipos de memória, com latências distintas. Logo, para além de alterações nas estruturas de dados, também é necessário ter em conta o local onde esses dados são armazenados.

Estes poderão ser alguns dos aspecto mais interessantes a desenvolver, ao nível do estudo dos Modelos de Iluminação Global por Radiosidade.

7 Anexos

Esta secção pretende completar alguns conceitos matemáticos usados ao longo do documento. Estes conceitos, na opinião do autor, são fulcrais para a compreensão de algumas matérias presentes neste documento.

7.1 Ângulo Sólido Diferencial

A posição na superfície de uma esfera pode ser dada por dois ângulos, um a partir do Pólo Norte ou zénite, θ, e outro à volta do equador, ϕ. Estes dois ângulos formam o sistema de coordenadas esféricas (θ, ϕ).

Se quisermos calcular a pequena área diferencial da superfície da esfera com raio r poderemos recorrer à seguinte equação:

$$dA = r^2 \sin\theta \, d\theta \, d\phi$$

onde $d\theta$ e $d\phi$ nos dão o deslocamento infinitesimal que gera a área.

O ângulo suportado por um arco circular de comprimento l é igual a l/r. O próprio círculo tem um ângulo de 2π devido à circunferência de um círculo ser $2\pi r$. Se extrapolarmos esta ideia podemos concluir que o ângulo suportado por uma área esférica a é igual a a/r^2. Esta quantidade é medida em *esterradianos* (radiano quadrados).

A área de um ângulo sólido diferencial ($d\omega$) é então dada pela seguinte equação:

$$d\omega = \frac{dA}{r^2} = \sin\theta \, d\theta \, d\phi$$

É muito conveniente pensar no ângulo sólido diferencial como um vector, em que a sua direcção indica o ponto na superfície da esfera e a sua norma o tamanho do ângulo sólido diferencial nessa direcção.

7.2 Norma de uma Função

A norma de uma função mede a magnitude de uma determinada função, muito semelhante à norma de um vector ou de um valor escalar. A equação que permite calcular a norma de uma função é seguinte:

$$L_p = \|\phi\|_p = \left[\int_{\phi} |\phi(x)|^p dx\right]^{1/p}$$

onde ϕ é o domínio da função.

As normas mais comuns são a L_1, L_2 e a L_∞ (substituído p por 1, 2 ou ∞). A norma L_1 aplicada ao estimador de erro $\varepsilon(x)$ resulta na área entre a função real e a aproximação. A norma L_2 é semelhante a L_1mas dá mais peso a erros maiores. A L_∞ é o erro máximo da região.

Como acontece com outras funções com integrais, esta tem de ser discretizada para viabilizar a sua utilização no hardware.

$$L_p(\phi) = \left[\sum_i |\phi(x_i)|^p \Delta x\right]^{1/p}$$

A variável Δx representa a área da superfície em estudo. Mesmo que esta seja subdividida, o erro global (o erro que contempla todas as superfícies de uma cena) poderá manter-se constante, uma vez que Δx está relacionado com as características da superfície (área).

8 Bibliografia

[1] Foley, James D.; Dam, Andries van; Feiner, Steven K.; Hughes, John F.; Phillips, Richard L., *Introduction to Computer Graphics,* s.l. : Addison Wesley, 1994.

[2] Dunn, Fletcher e Parberry, Ian, *3D Math Primer for Graphics and Game Development,* Texas : Worldware Publishing, Inc., 2002.

[3] Gouraud, Henri, *Continuous Shading of Curved Surfaces,* s.l. : CS Digital Library, 1971.

[4] Phong, Bui Tuong, *Illumination for Computer Generated Pictures,* 1975.

[5] "Cornell Box Data," *Cornell University Program of Computer Graphics,* Cornell University, 02 de Fevereiro de 2005, retrieved 9 de Julho de 2009, http://www.graphics.cornell.edu/online/box/data.html.

[6] Dubuis, Eric e Bieri, Hanspeter, *Dynamic Subdivision in Radiosity,* Bern, Switzerland : University of Bern, 1994.

[7] Portela, César, *Light and Architecture,* La Palma : StarLight, 2007.

[8] Kessenich, John, Baldwin, Dave e Rost, Randi, *The OpenGL® Shading Language,* s.l. : 3Dlabs, Inc. Ltd., 2006.

[9] Cohen, Michael F e Wallace, John R, *Radiosity and Realistic Image Synthesis,* Cambridge : Academic Press, 1993. ISBN 0-12-178270-0.

[10] "Light," *Wikipedia.org,* 5 de Agosto de 2007, retrieved 20 de Julho de 2009, http://en.wikipedia.org/wiki/Light.

[11] "Lambert's cosine law," *Wikipedia.org,* 9 de Janeiro de 2009, retrieved 28 de Março de 2009, http://en.wikipedia.org/wiki/Lambert%27s_cosine_law.

[12] Spencer, Stephen, *Education Slide Set,* [Diapositivo] s.l. : SIGGRAPH, 1993.

[13] Gortler, Steven, Cohen, Michael F. e Slusallek, Philipp, *Radiosity and Relaxation Methods: Progressive Refinement is Southwell Relaxation,* Department of Computer Science, Princeton University : s.n., 1993.

[14] Munshi, Aaftab, [ed.], *The OpenCL Specification,* s.l. : Khronos OpenCL Working Group, 2008.

[15] Heidrich, Wolfgang e Seidel, Hans-Peter, *View-independent Environment Maps,* s.l. : SIGGRAPH / Eurographics Workshop on Graphics Hardware, 1998. pp. 39-46, Lisboa, Portugal.

[16] Neumann, Laszlo, Matkovic, K e Purgathofer, Werner, *Automatic Exposure in Computer Graphics Based on the Minimum Information Loss Principle,* Washington : IEEE Computer Society, 1998. ISBN 0-8186-8445-3.

[17] Debevec, Paul Ernest e Malik, Jitendra M, *Recovering high dynamic range radiance maps from photographs,* Los Angeles, California : ACM, 2008.

[18] Jacobson, Ralph; Ray, Sidney; Attridge, Geoffrey G; Axford, Norman, *Manual of Photography: Photographic and Digital Imaging,* Burlington : Focal Press, 2000. ISBN 0-240-51574-9.

[19] Barnes, Mark e Finch, Ellen Levy, *COLLADA - Digital Asset Schema Release 1.5.0,* [Specification] s.l. : Sony Computer Entertainment Inc., 2008.

[20] OpenMP Architecture Review Board, *OpenMP Application Program Interface Version 3.0,* [Specification] 2008.

[21] Hasenfratz, Jean-Marc; Damez, Cyrille; Sillion, François; Drettakis, George, *A Practical Analysis of Clustering Strategies,* Grenoble, France : EUROGRAPHICS '99, 1999.

[22] Ventura, Hugo e Próspero dos Santos, Manuel, "Simulação e Controlo do Efeito de Profundidade de Campo em Tempo de Interacção," s.l. : EPCG-GPCG, 2009, pp. 93-102.

[23] Ventura, Hugo, *Simulação da Profundidade de Campo, Dissertação de Mestrado em Engenharia Informática,* FCT-UNL, 2009.

[24] Pharr, Matt e Fernando, Randima, *GPU gems 2: Programming Techniques for High-performance Graphics and General-purpose Computation,* s.l. : Addison-Wesley, 2005. pp. 635-648, ISBN 0-3213-3559-7.