

MODELAGEM PARALELA EM ANEL DO "RAY TRACING"

DANIEL CÂMARA

ORIENTADOR PROF. ANDRÉ L. PIRES GUEDES

Universidade Federal do Paraná
Setor de Ciências Exatas
Departamento de Informática - Dinf
Centro Politécnico - Jardim das Américas
Cep 81531-990 Curitiba - Paraná
Telefones: (041)367-1421
(041)366-2323 R. 3463
daniel@cce.ufpr.br
andre@inf.ufpr.br

Abstract: Ray tracing is an algorithm to generate realistic images from three dimensional scenes. The main goal of this paper is to give a parallel alternative to calculate the ray tracing, improving the traditional form, which is inefficient. We're going to show some ways to paralelize the algorithm and our solution, which consists of the dynamic division of the image plane among process organized in a structure similar to a ring network.

Keywords: Ray-Tracing, cost, atom.

Resumo: Ray tracing é um algoritmo para gerar imagens realísticas a partir de cenas tridimensionais. O objetivo deste artigo é fornecer uma alternativa paralela para o cálculo do "ray tracing", já que este, em sua forma normal de cálculo, se revela ineficiente. Mostraremos algumas formas de paralelização do cálculo e apresentaremos nossa solução, que consiste na divisão dinâmica da imagem entre processos organizados em uma estrutura que lembra uma rede em anel.

Palavras Chave: Ray Tracing, custo, átomo.

1 - Introdução

O "Ray Tracing" é um algoritmo para a produção de imagens ou figuras realistas. Introduzido em sua forma revisada por Whitted [Whitted (1980)], permite a obtenção de imagens com alto grau de realismo. Apresenta, entretanto, algumas desvantagens, como o alto custo computacional e a necessidade de uma grande quantidade de memória. Várias alternativas foram propostas para contornar estes problemas [Corrêa (1993)], utilizaremos a paralelização como forma de aceleração do processo.

2 - Cálculo do "Ray Tracing"

O cálculo do Ray Tracing, em um alto grau de abstração, consiste em calcularmos o segmento de reta que passa pelo observador e pelo ponto do plano de observação, e descobriremos qual objeto mais próximo do observador é interceptado por esta reta. A partir do ponto de intersecção, calcula-se a quantidade de luz que este ponto recebe das fontes de luz, e quanto ele recebe de luz refletida e refratada por outros objetos. Tem-se então dois novos raios que devem ser seguidos (luz refletida e refratada), gerando desta forma uma árvore de raios a calcular. Este cálculo termina quando: a contribuição de luz do raio é irrelevante, a árvore de raios chega em um nível que consideraremos como limite, ou o raio não intercepta nenhum outro objeto da cena. A soma dos coeficientes fornece a cor do ponto.

3 - Divisão Bidimensional

É importante que façamos uma distinção entre cena, que são os objetos do mundo real, e imagem, que é a representação desta cena em um plano bidimensional. Uma das formas de fazermos a paralelização do Ray Tracing é a partir da divisão da imagem. De forma simplista, dividiremos a paralelização em divisão simples e divisão complexa.

Divisão Simples - É o processo pelo qual dividimos o cálculo da imagem pelo número exato de processadores disponíveis, uma divisão estática. Neste método, algum processador pode ficar sobrecarregado, já que as partes da imagem não têm necessariamente a mesma complexidade. Se pensarmos em uma rede de computadores, o processo torna-se um pouco pior, pois distribuimos, além das partes da imagem, também a cena. Podemos ainda ter problemas com máquinas mais lentas, ou com um custo de rede alto, o que subutilizaria os outros computadores.

Divisão Complexa - Podemos também dividir a imagem em mais partes que

processadores disponíveis. O processo pai divide a imagem e envia um pedaço desta juntamente com a cena para cada filho. Este, ao acabar de calcular a parte que lhe foi dada, envia os resultados ao pai, requisitando outra parte. O filho recebe uma nova parte ou uma mensagem avisando que ele não é mais necessário.

Esta forma faz uma distribuição dinâmica dos pacotes, melhorando bastante a questão do aproveitamento das máquinas. Podemos ainda fazer um cálculo sobre as requisições de novas áreas, detectando máquinas *provavelmente* mais lentas da rede. Conseguimos assim fazer um balanceamento dinâmico das cargas das máquinas, porém com um custo: o aumento de comunicação na rede.

4 - Modelo Bidimensional em anel

Nossa proposta é uma melhoria sobre as duas anteriores. Dividimos a imagem exatamente entre o número de máquinas disponíveis e cada filho subdivide sua parte em N partes iguais que chamaremos de átomos, estes não serão mais divididos. Cada filho cria um novo processo para o cálculo dos átomos, um neto. Este recebe através de um pipe o número do átomo que deve calcular. Ao término deste cálculo, envia os resultados ao processo que o criou (filho), onde vão sendo armazenados.

Quando um processo termina a faixa da imagem que lhe foi dada calcular, envia ao pai os resultados e uma oferta de ajuda ao seu irmão da esquerda (com o número de átomos que calculou). O irmão compara o número de átomos calculados com o que ele próprio já calculou, podendo desta forma fazer uma média da velocidade do irmão baseado na sua própria velocidade. Decide assim quantos átomos deve pedir ao irmão que calcule ou, se não for vantajosa a oferta, passa o pedido ao irmão da esquerda (Fig. 1). Dessa forma é feito um balanceamento dinâmico da carga de cálculo.

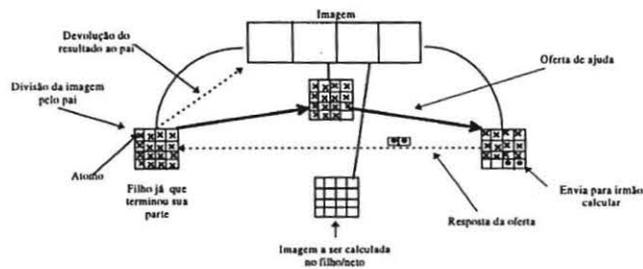


Fig. 1 - Descrição de como funciona o cálculo de uma cena paralelizada pela nossa solução.

Os filhos ficam organizados em um anel e a mensagem de oferta de ajuda funciona como um token [Baudouel (1994)]. Se algum irmão precisa de ajuda, consome

o token e envia os átomos a calcular diretamente ao irmão que fez a oferta.

A função do filho acaba quando enviar um pacote de oferta para o irmão da esquerda e receber o *mesmo* pacote de seu irmão à direita, indicando que nenhum irmão precisou de ajuda. Neste momento a imagem já deve estar completa, ou quase completamente calculada, ficando o filho à espera de uma mensagem de suicídio do pai.

5 - Conclusões

O algoritmo ocupa mais rede que o primeiro e menos que o segundo. Seu desempenho é melhor que o segundo em termos de recursos computacionais, pois o computador não fica parado esperando as partes que deve calcular (os átomos ficam enfileirados no pipe). Não tem também o gargalo que seria o pai no controle da distribuição das partes da imagem. A perda com ociosidade é mínima, se for bem implementada a granularidade dos átomos e a comunicação dentro do anel.

Mesmo se ocorrer de uma máquina da rede ser muito lenta e outras rápidas não é problema, a máquina lenta não irá “atrapalhar”, já que seus irmãos virão em sua ajuda. Se a máquina lenta consumir todas as ofertas de ajuda ainda assim não será problema, pois todas as máquinas estarão ocupadas, e ao término do cálculo destinado à máquina mais lenta, estarão disponíveis a ajudar os outros irmãos. Mesmo a máquina mais lenta não ficará subutilizada, já que os irmãos saberão de sua baixa capacidade sempre alocarão para suas ofertas, uma quantidade de átomos compatível ao seu desempenho. O cálculo é feito visando que todos os processos terminem ao “mesmo tempo”. O método não é restritivo, ou seja, permite a utilização de outras técnicas para a melhoria do desempenho do Ray Tracing.

6 - Referências

T., Whitted “An improved illumination model for shaded display”, Communications of the ACM Vol 23(6) pags 343-349, 1980

W. T. Corrêa, M. A. de C. Lima, W. M. Júnior, M. L. B. De Carvalho, “ Aceleração de Ray Tracing via Heterogeneidade” SIBIGRAPI'93 - VI Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens. pags. 1-8, 1993

Didier Baudouel, Bouatouch, Kadi Bouatouch, Thierry Priol, “Distributing Data and Control for Ray Tracing in Parallel”, IEEE Computer Graphics and Applications, July 1994, pags. 69-77