6
Conclusão

Foi apresentado nesta dissertação um estudo detalhado da técnica de mapeamento de textura com relevo e foram implementadas as seguintes variantes:

Amostragem 1D em dois passos. A amostragem unidimensional de texturas com relevo realizada em dois passos é a abordagem mais simples apresentada em [26]. No entanto, esta variante é propensa ao aparecimento de ruídos resultantes de oclusão e interpolação.

Amostragem assimétrica em dois passos. Ruídos resultantes da interpolação na amostragem unidimensional causam distorções não-lineares (por exemplo, linhas retas na textura original aparecem curvadas na imagem resultante). O efeito não-linear da interpolação pode ser corrigido substituindo-se a interpolação e o armazenamento de valores de profundidade pela interpolação e o armazenamento das coordenadas da linha durante o primeiro passo. Tais valores de linha que antes eram calculados no segundo passo agora são calculados durante o primeiro passo, daí o nome amostragem assimétrica.

Amostragem com compensação de deslocamento em dois passos. Nesta variante, calculam-se valores de profundidade mais precisos durante o primeiro passo para que valores de coordenadas almejados sejam obtidos durante o segundo passo.

Amostragem intercalada em um passo. Para os casos nos quais a sobreposição de *texels* durante o primeiro passo poderia causar auto-oclusão, o processo de reconstrução realizado em dois passos pode ser substituído por um processo que intercale ambos os passos num único passo. Além disso, esta nova abordagem não requer comparação de valores de profundidade para resolver o problema da auto-oclusão.

PUC-Rio - Certificação Digital Nº 0220939/CA

Pode-se concluir que a amostragem intercalada em um passo é a variante mais eficiente em termos de fps apesar de ter um desvio padrão mais elevado. A variante mais estável, isto é, com menor desvio padrão é a amostragem 1D em dois passos.

Além disso, com o intuito de otimizar o processo de amostragem e reconstrução a partir de texturas com relevo, foram implementadas duas novas abordagens:

Paralela. Na abordagem paralela criou-se uma *thread* de CPU para executar os quatro primeiros passos do algoritmo de mapeamento de textura com relevo. Tal *thread* é executada em paralelo com o processo principal da CPU, utilizando-se para este fim a tecnologia de *Hyper-Threading*.

Multi-Threaded. Nesta abordagem, dividiu-se a textura de entrada em duas partes com o intuito de executar, simultaneamente, os quatro primeiros passos do algoritmo de mapeamento de textura com relevo para cada uma destas partes, sendo necessário um pós processamento para unir os dois resultados obtidos numa única textura resultante.

É possível concluir que, para uma câmera estática, a abordagem paralela é superior à abordagem convencional (seqüencial). Para casos específicos em que o número de *texels* inválidos é muito pequeno ou nulo, a abordagem *multi-threaded* é superior à abordagem seqüencial. Já, para uma câmera em constante movimento, há uma queda na taxa de quadros por segundo fazendo com que a abordagem *multi-threaded* seja menos eficiente que a abordagem seqüencial.

Além disso, acrescentado-se a informação de normal a uma textura com relevo e realizando um cálculo de iluminação por *pixel*, é possível aumentar o realismo dos resultados obtidos com a técnica de mapeamento de textura com relevo.

Como visto no Capítulo 5, a taxa na qual as imagens são geradas é medida em quadros por segundo (fps). Em um quadro por segundo, existe muito pouco de interatividade; o usuário espera impacientemente pela chegada de cada nova imagem. Por volta de 6fps, a noção de interatividade começa a crescer. Uma aplicação exibindo imagens a 15fps é certamente uma aplicação de tempo real. No entanto, existe um limite para tal interatividade. De aproximadamente 72fps em diante, diferenças na taxa de atualização são efetivamente imperceptíveis.

PUC-Rio - Certificação Digital Nº 0220939/CA

Dessa forma, de acordo com as observações anteriores e os resultados descritos no Capítulo 5, é possível concluir que a técnica de mapeamento de textura com relevo é perfeitamente viável de ser utilizada em aplicações com requerimento de tempo real, em particular, jogos eletrônicos. Além disso, como observado por Oliveira [26], é possível que este algoritmo seja implementado em *hardware* e, futuramente, suportado pelas placas gráficas, o que tende a aumentar o seu desempenho computacional consolidando, ainda mais, a conclusão alcançada através da realização desta dissertação.

6.1
Trabalhos Futuros

A técnica de renderização baseada em imagem surgiu a partir de uma combinação entre as técnicas de Síntese 3D de Imagem e Visão Computacional, tendo em vista adquirir resultados foto-realistas e ao mesmo tempo rápidos na visualização de cenas 3D [11].

Para uma definição mais formal do que vem a ser renderização baseada em imagem, deve-se recorrer ao conceito de *função plenóptica* (do inglês, *plenoptic* onde *plenus* = completa e *optic* = visão), definida inicialmente por Adelson & Bergen [1]. Esta é uma função parametrizada que descreve tudo o que é possível de ser visto a partir de qualquer parte do espaço, a qualquer momento e em qualquer comprimento de onda. Em [25] tal função é definida como:

$$p = P\left(\theta, \phi, \lambda, v_x, v_y, v_z, t\right)$$

onde θ e ϕ são os ângulos de azimute e elevação da direção de visão do observador, λ é o comprimento de onda da luz a ser observada, (v_x, v_y, v_z) é a posição do observador e, finalmente, t é a dimensão de tempo para o caso de cenas animadas. Veja a figura 6.1.

Desse modo, uma possível definição para a maioria dos problemas de Renderização Baseada em Imagem pode ser descrita da seguinte forma: ***"Dado um conjunto de amostras (completo ou incompleto) da função plenóptica, o objetivo da renderização baseada em imagem é gerar uma representação contínua desta função"*** [25].

Porém, devido à impossibilidade de implementação da função plenóptica no seu caso geral, os sistemas de Renderização Baseada em Imagem procuram criar amostras da mesma a partir de um conjunto finito e

PUC-Rio - Certificação Digital Nº 0220939/CA

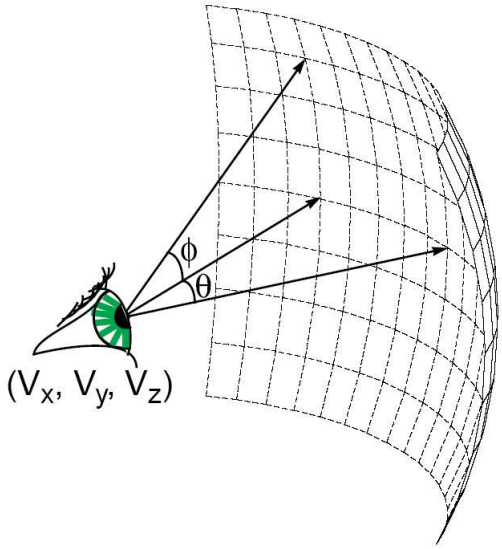

Figura 6.1: Qualquer ponto visível de qualquer posição de um observador pode ser descrito pela função plenóptica. (*Imagem obtida em [25]*)

PUC-Rio - Certificação Digital Nº 0220939/CA

discreto de dados.

Na prática, todos os métodos de visualização baseados em imagem possibilitam a manipulação de sub-conjuntos de variáveis (menor ou igual a sete) da função plenóptica. Quanto maior o número de variáveis livres maior será o grau de liberdade permitido na interatividade do sistema implementado. Dessa maneira, por exemplo, restringindo-se o problema para cenas estáticas capturadas em comprimentos de onda discretos (exemplo, vermelho, verde e azul), é possível reduzir a função plenóptica para cinco dimensões. Este é o caso da técnica de mapeamento de textura com relevo.

Uma possível extensão para o trabalho apresentado nesta dissertação é derivar uma parametrização da função plenóptica 6D conveniente para aplicações que suportam *walkthrough* em cenários onde o tempo transcorra linearmente, permitindo-se que diversos objetos estejam movendo-se individualmente. Em outras palavras, esta nova parametrização deve suportar a visualização de superfícies animadas.

Além disso, duas questões de otimização que podem resultar em trabalhos interessantes são a utilização de um *pipeline* de multi-processadores no processo de amostragem e reconstrução, e a uma variante da abordagem *multi-thread* que processe as linhas e colunas da textura em paralelo.

Com o uso da tecnologia de *Hyper-Threading*, é possível montar um *pipeline* na CPU composto de dois estágios, um para cada processador disponível. O primeiro estágio seria responsável por realizar os três primeiros

passos do algoritmo de mapeamento de textura com relevo mais o *warping* horizontal. O segundo estágio seria responsável por realizar o *warping* vertical juntamente com o restante das operações necessárias.

Em relação à variante da abordagem *multi-threaded*, o algoritmo de amostragem e reconstrução seria realizado através de três processos P_1, P_2 e P_3. O processo principal P_1 seria responsável por gerenciar a distribuição de tarefas entre os dois processos restantes da seguinte maneira: durante o passo horizontal do pré *warping*, inicialmente, a primeira linha da textura de entrada é enviada para o processo P_2 enquanto que a segunda linha é enviada para o processo P_3. A partir daí, o processo P_1 detém um ponteiro para a próxima linha a sofrer uma operação de *warping*. Desse modo, assim que P_2 ou P_3 terminar a respectiva operação de *warping*, uma nova linha da textura pode ser requisitada ao processo P_1. Este ciclo ocorre até que não haja mais linhas a serem processadas. O mesmo ocorre durante o passo vertical, porém, ao invés das linhas são processadas as colunas da textura de entrada. A vantagem deste método em relação à abordagem *multi-threaded* apresentada no Capítulo 4 é que o mesmo não precisa realizar o pós processamento necessário para unir os dois resultados obtidos numa única textura resultante, o que certamente ocasionará uma melhora considerável no tempo de processamento consumido.

Além disso, um estudo sobre a taxa de *warpings* por segundo (wps, ou seja, o número de operações de *warping* realizadas por segundo) em relação à taxa de quadros por segundo poderia permitir uma implementação mais eficiente (com menos operações de *warping*) do processo, caso a taxa de wps se confirme ser maior que a taxa de fps.

Finalmente, uma estratégia de implementação que utilize vários processadores físicos, ao invés do uso da tecnologia de *Hyper-Threading*, seria um trabalho interessante para se comparar com as abordagens aqui implementadas.

PUC-Rio - Certificação Digital Nº 0220939/CA