9 Conclusão

I don't know the future. I didn't come here to tell you how this is going to end. I came here to tell you how it's going to begin. (The Matrix)

9.1 Contribuições

Foi apresentado que através da técnica de impostores com relevo é possível aumentar consideravelmente o tempo de vida de um objeto sendo representado por uma imagem. Diferentemente do método tradicional dos *sprites*, entretanto, faz-se necessário fornecer ao sistema de visualização a malha do modelo a ser utilizado, bem como o algoritmo de renderização. Outra alternativa que se pode seguir para atingir tal fim consiste nos *Layered Depth Images*, mas destaca-se que os impostores com relevo são mais simples de serem gerados dinamicamente, por se adaptarem facilmente aos algoritmos de visualização padrão.

Mostrou-se que os impostores com relevo podem representar qualquer geometria, estendendo a capacidade de modelagem por texturas com relevo, apresentado inicialmente por Oliveira (1999).

No trabalho, pode-se comprovar também que em alguns casos é mais conveniente dividir o processamento de um sistema de *ibr* em duas etapas (uma para a CPU e outra para a GPU) do que implementá-la por completo em *hardware*. Isto porque o processo de *warping* inverso possui um inconveniente que é a falta de conhecimento da profundidade do *pixel*.

A complexidade inerente à visualização de texturas com relevo é independente da complexidade da geometria do objeto representado. No caso da implementação de (Policarpo, 2003), para um grau de realismo razoável este processamento é demorado, especialmente devido as limitações impostas pelos recursos de *hardware*. Já no caso dos impostores com relevo, apesar da velocidade para atualizar as texturas ser dependente da geometria e do algoritmo de visualização sendo utilizado, comprovou-se que há uma grande eficiência na

atualização. Mais ainda: para o *hardware* disponível na época da elaboração deste trabalho, o desempenho foi muito superior para esta segunda abordagem. Isto porque o *hardware* fica dedicado a tarefas mais simples, adequadas à sua capacidade atual.

Comprovou-se também nesta tese, em colaboração com (Fonseca, 2004) que é possível paralelizar o processo de mapeamento de texturas com relevo a diversos níveis: CPU/GPU, *multi-threading* e multi-processamento. Os resultados de algumas abordagens são apresentados nos capítulos 6 e 7.

Finalmente, uma importante contribuição da tese consiste na construção de um *framework* que permite inserir elementos geométricos e impostores com relevo com bastante facilidade. Este sistema simplifica consideravelmente a continuação da pesquisa aqui iniciada.

9.2 Trabalhos Futuros

Uma possível extensão ao que foi discutido no capítulo 7.8 consiste em criar um sistema de previsão para gerar impostores com relevo: quando o *thread* dedicado a gerar novas texturas com relevo estiver ocioso, este poderia ser aproveitado para gerar texturas para futuras posições do observador, armazenando estes resultados no *cache*, enquanto ainda não sejam requisitados. Este sistema de previsão pode ser baseado numa estimativa sobre futuras posições do observador, tendo em vista a análise do vetor velocidade. O gerenciamento do *cache* pode possuir certo grau de “inteligência”, procurando descartar texturas para posições menos prováveis em que o observador pode vir a se encontrar, quando se torna necessário liberar espaço de memória.

Para a tarefa de gerar dinamicamente novas texturas com relevo, pode-se desenvolver um sistema de cálculo progressivo. Desta forma, no caso de uma textura ser solicitada mas esta ainda não ter sido gerada por completo, envia-se ao *pipeline* a textura com uma resolução menor porém completa.

Neste trabalho apenas se discutiu sobre a utilização de impostores com relevo estáticos. Uma extensão da pesquisa deve considerar objetos animados. Como sugerido em (Oliveira, 2000), a translação do impostor com relevo pode ser feita com uma simples translação do polígono de suporte, com possíveis atualizações da textura, de acordo com a métrica apresentada; a rotação também

consiste num movimento deste tipo no polígono de suporte e a escala também pode ser feita com uma simples transformação do polígono, levando em conta a utilização de texturas com relevo com multiresolução. Neste caso, um sistema de previsão pode ser muito importante, pois ao contrário de que uma previsão de posições do observador, onde não há certeza, a previsão de transformações provindas de uma animação já existente possui 100% de acerto. Esta abordagem corresponderia a inserir o parâmetro tempo na função plenóptica, com a limitação de animações poderem ser apenas de translação, rotação e escala.

Para o caso de não existir o mapa de normais para a textura com relevo em questão, a seção 5.6 apresenta uma forma de criar valores estimados. Para esta finalidade, pode-se criar uma modelagem mais complexa, no entanto mais realista, realizando uma pré-avaliação e um reconhecimento de padrões da imagem.

Ao gerar dinamicamente as texturas com relevo, é possível fazer uma abordagem onde se acrescenta para cada *texel* o cálculo de sombreamento. Isto pode ser particularmente útil quando as luzes do cenário e os objetos forem sempre estáticos, ou para sistemas que não dispõem de *fragment processors* na GPU.

Dentro de um ambiente de visualização 3D, como num *engine* gráfico para jogos 3D, por exemplo, poderia-se desenvolver critérios e métodos inteligentes de identificar quais objetos devem ser tratados através de geometria e quais através de imagem (no caso, impostores com relevo). Pode ser que um mesmo objeto, num espaço de tempo deva ser visto como geometria e em outro momento deva ser tratado como imagem. Para isto, deveria-se analisar como realizar um processo de transição suave.

Uma possível aplicação dos impostores com relevo poderia dar-se em sistemas de visualização distribuídos por *hardware*: dentro de uma rede constata-se que se uma CPU distribui *frames* para serem calculados por várias GPU's, em máquinas separadas, devido à velocidade de transmissão de dados da rede, há um tempo de atraso grande na devolução das imagens para o servidor. Este tempo impede que se possa manter uma taxa de *frames* por segundo adequada para um sistema de visualização em tempo real. Uma possível paralelização seria enviar às demais GPU's requisições de possíveis impostores com relevo que serão necessários para os objetos do cenário. Isto atenuaria o atraso, uma vez que o impostor com relevo tem um tempo de vida maior que um impostor simples.

Utilizando-se o sistema de previsão discutido, poderia-se aliviar grandemente a CPU/GPU principal.

Por fim, uma interessante pesquisa consistiria na implementação de impostores com relevo através do algoritmo de *view-morphing*, ao invés do 3D *image warping*. Neste caso os objetos deveriam ser chamados de impostores com *morphing*...