Departamento de Matemática

VISUALIZAÇÃO E RECONSTRUÇÃO PARA NUVEM DE PONTOS

Aluno: Alexandre Marangoni Costa
Orientador: Sinésio Pesco

Introdução

Nuvens de pontos (point-clouds) surgem principalmente como resultado de saída de dispositivos como scanners 3D e vêm obtendo grande importância nos últimos tempos, principalmente devido ao grande volume de dados a serem processados.

Um problema relevante a ser resolvido é a visualização de uma nuvem de pontos por um observador (câmera), visto que um ponto p1, em geral, não oculta outro ponto p2 situado atrás dele (a menos que p1, p2 e o observador sejam colineares). Esta característica faz com que a visualização de um objeto por uma nuvem de pontos fique bastante dúbia. Assim sendo, este trabalho propõe uma implementação que visa otimizar esta visualização. A referência teórica para o desenvolvimento desta implementação é o trabalho de Katz et al.[1] que introduziu uma técnica de visualização de nuvens de pontos bastante eficiente, além de ter baixo esforço computacional se comparada a outras técnicas.

Conceitos Básicos

Fecho convexo. Dado um conjunto de pontos P, um fecho convexo é uma combinação finita de elementos desse conjunto de modo a formar uma figura convexa.

Neste trabalho, o algoritmo para a aplicação do fecho convexo foi extraído do livro "Computational Geometry in C" (segunda edição) de Joseph O'Rourke [2] com algumas modificações para melhor se adequar ao nosso propósito.

Reflexão sob a superfície de uma esfera. Considere uma esfera de raio R, centrada em C, que contenha todos os pontos de um conjunto P. Pode-se refletir um ponto $p_i \in P$ sob a superfície desta esfera a partir da seguinte equação:

$$p_i = f(p_i) = p_i + 2 \cdot (R - \|p_i\|) \cdot \frac{p_i}{\|p_i\|}$$

O objetivo da reflexão é suavizar as concavidades do objeto quando visto por um observador localizado no centro da esfera.

Descrição do Método

Este método foi desenvolvido primariamente por Katz et al. [1] , e, posteriormente, foi implementado em OpenGL usando C, e explorado neste trabalho.

Inicialmente tem-se uma point-cloud P1. Aplicando uma reflexão esférica sobre P1, obtêm-se como resultado outra point-cloud P2. A principal característica da point-cloud P2 é que, nela, observa-se a suavização da região mais próxima ao observador, que reduz as concavidades, e, além disso, como o objeto foi refletido, esta superfície encontra-se, agora, no lado oposto ao lado visto pelo observador.

Para selecionar os pontos desejados, calcula-se o fecho convexo sobre o conjunto de pontos de P2 (nuvem refletida) e inclui-se o ponto que representa a posição do observador. Verifica-se nesta etapa a importância da etapa anterior, pois se a reflexão não fosse feita, a superfície desejada seria côncava, o que impediria o fecho convexo de selecionar todos os seus pontos.

Como resultado, os pontos que pertencem ao fecho são os pontos visíveis ao observador, antes da reflexão.

Descrição do Código

Uma nuvem de pontos, geralmente, é guardada num arquivo de texto. Este arquivo, guarda a informação de quantos pontos compõem certa nuvem de pontos, e, posteriormente, fornece as coordenadas de cada ponto.

Visto isso, o código inicia-se com a leitura do arquivo de texto, guardando número total de pontos da point-cloud. Esta informação é utilizada para fazer a alocação dinâmica de uma matriz M[n][3], que irá guardar as coordenadas dos pontos da nuvem, onde n é o número de pontos e 3 são as coordenadas x, y e z. De posse desta matriz, os dados são armazenados nela.

A partir daí, os métodos anteriormente explicados são aplicados da seguinte forma: a matriz criada é utilizada para gerar outra matriz G[n][3], que armazenará os pontos da matriz M refletidos esfericalmente e, em seguida, a matriz G e a posição do observador são submetidas a aplicação de um fecho convexo. O algoritmo de fecho convexo dará ao programa, como saída, o índice dos vértices dos pontos capturados pelo fecho.

De posse destes índices, utiliza-se a biblioteca gráfica OpenGL para localizá-los na matriz M e plotá-los, e, enfim, teremos uma imagem bem mais nítida da point-cloud.

Resultados

A implementação utilizada neste trabalho mostra-se eficiente para vários tipos de point-clouds. Essa eficiência depende, porém, da manipulação de parâmetros que devem variar com o espaçamento entre os pontos utilizados, bem como com o tamanho virtual do objeto (maior distância calculada entre dois pontos deste). Ainda assim, os resultados são bastante satisfatórios (como pode ser conferido nas figuras abaixo), deixando a visualização da nuvem de pontos bem mais nítida e com menor esforço computacional.

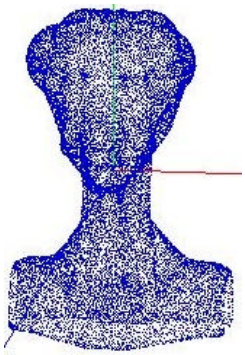

fig. 1: Original

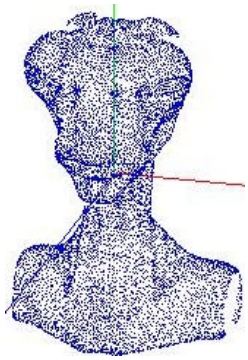

fig. 2: Após o método

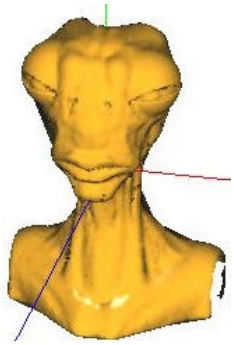

fig. 3: Aplicação de iluminação

Referências

[1] Katz S., Tal A., and Basri R., Direct Visibility of Point Sets, *SIGGRAPH 2007, ACM Transactions on Graphics*, Volume 26, Issue 3, August 2007

[2] J. O'Rourke, Computational Geometry in C, Cambridge University Press, 1994. Segunda edição, 1998.

Jack E. Bresenham, "Algorithm for computer control of a digital plotter", *IBM Systems Journal*, Vol. 4, No.1, January 1965, pp. 25–30

M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, C. Silva. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics* 9(1):3–15, 2003.