MC930 – Computação Gráfica - 2007-S1 - Jorge Stolfi
Trabalho de laboratório 12: Busca de Árvores Eficientes

Nome	**RA**	**Nota**

Objetivos: Treinar o uso de macros recursivas e (pseudo-)fractais.
Enunciado: Apesar do aspecto antiquado, limitada gama de cores e documentação precária, árvores têm certas utilidades — como pendurar cartazes, impedir a passagem de veículos ou pedestres, e fomentar a economia nacional nos setores de limpeza urbana e jardinagem. Mas, sem dúvida, elas seriam muito mais úteis (e mais populares entre arquitetos e urbanistas) se tivessem formas mais simples e mais eficientes em termos de espaço ocupado.

Posta essa premissa, sua missão para hoje é desenhar uma *árvore eficiente*, semelhante a uma árvore tradicional mas cujo formato externo é um sólido simples que aproveita bem o espaço. Exemplos ideais são sólidos que podem preencher todo o espaço $\mathbf{R}^3$ sem deixar vãos, como cubo, o prisma regular de base hexagonal ou o dodecaedro romboidal. O cilindro e a esfera não são ideais, mas ainda são aceitáveis. Já o cone ou toro são inadequados sob esse critério.

Uma maneira relativamente simples de desenhar uma árvore é observar que ela tem uma estrutura recursiva. Ou seja, cada galho (em particular, a arvore toda) consiste de um "tronco" cilindrico, na extremidade do qual estão fixados dois ou mais galhos menores. A base dessa recursão é uma folha, flor, fruto, botão, etc..

Parte 1. Antes de comecar a programar, desenhe (á mão livre, na folha providenciada) um esboço de sua árvore, indicando o sólido geométrico simples que a contém. Desenhe também o objeto que é a base da recursão (folha, fruto, etc. — mas não complique!)

Parte 2. Gere uma imagem da sua árvore com POV-Ray. Para isso, escreva uma macro recursiva `galho`(p, v, L, r) que gera uma descrição POV-Ray de um galho da sua árvore. Nessa macro, p é um ponto, o centro da base do galho; v é um vetor de comprimento unitário, que dá a direção do tronco; L é o comprimento do tronco; e r é o raio do tronco. A chamada inicial pode ser, por exemplo,

```
galho(<0,0,0>, <0,0,1>, 3, 0.2)
```

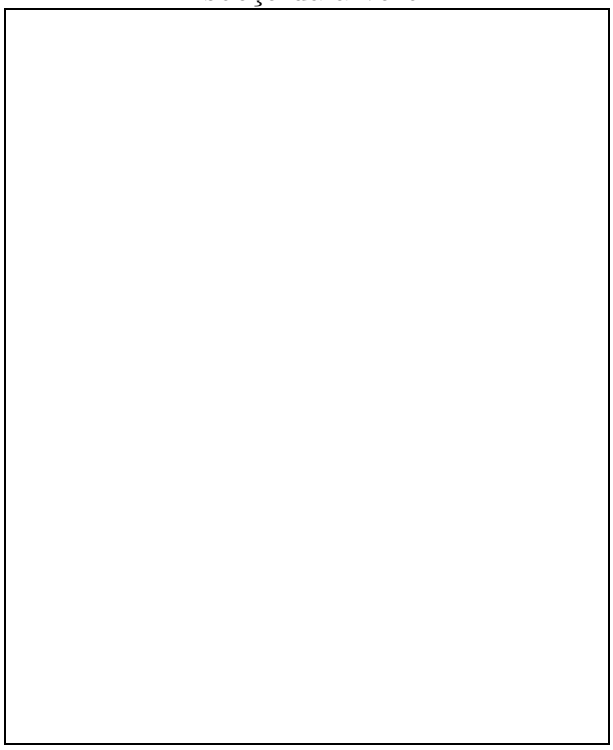
A macro deve gerar um objeto `union{}`, contendo o tronco do galho, que vai do ponto p ao ponto $q = p + L * v$; e dois sub-galhos com base no ponto q, gerados por chamadas recursivas da mesma macro.

Os sub-galhos devem ter comprimentos L' e L'', menores que L, e raios r' e r'', também menores que r. Uma maneira de obter os vetores de direção v', v'' correspondentes é somar ao vetor v dois vetores aleatórios u', u'', com coordenadas entre -0.5 e $+0.5$, e ajustar o vetor resultante para comprimento unitário. (A função `vnormalize`(w) do POV-Ray faz isso).

A recursão deve terminar quando o raio r ficar muito pequeno, ou quando o ponto q cair fora do sólido geométrico (cubo, cilindro, etc.) correspondente ao formato escolhido para a árvore. Para testar essa condição, note que as coordenadas de um ponto u podem ser obtidas

em POV-Ray com a notacão $u.x$, $u.y$, e $u.z$. Na base da recursão, a macro deve gerar uma folha ou similar, em vez do tronco e sub-galhos.

Esboço da árvore



Detalhe – base da recursão