

MC937A/MO603A – Computação Gráfica - 2020-S2 - Jorge Stolfi

Trabalho de laboratório 04 - 2020-10-28

Robogarçom

Objetivos: treinar uso de *modelos articulados hierárquicos*, e transformações `rotate` e `scale`.

Enunciado. O distanciamento seguro de clientes de um restaurante pode ser conseguido eapagando as mesas; porém, ao ir de mesa em mesa, o garçom continua sendo um grande risco de contágio do COVID-19. Máscaras ajudam, mas para segurança aceitável o garçom precisa ficar a pelo menos 2 metros de qualquer mesa, e servir pratos e bebidas por meio de um robô telecomandado com longos braços articulados.

Sua tarefa hoje é projetar tal robô, usando a técnica de *modelo articulado hierárquico*, a ser explicada na aula. Seu robô precisa ter pelo menos dois braços, cada um com no mínimo 4 graus de liberdade internos.

Parte 1. Antes de começar a programar, **desenhe um esboço do seu projeto e apresente-o ao professor via Meet, quando solicitado, no início da aula.**

Parte 2. Produza uma imagem da sua cena usando POV-ray. O modelo deve consistir de um conjunto de macros do POV-Ray `falangeta(...)`, `dedo(...)`, `mao(...)`, `braco1(...)`, `robo(...)` que geram os sub-componentes (sub-árvores) do robô. Cada macro gera sua componente numa posição e orientação fixas, com o ponto de ancoragem na origem.

Cada grau de liberdade *interno* da componente deve ser um parâmetro separado da macro. Por exemplo, se um dedo tem uma articulação interna, a macro `dedo` deve ter 1 parâmetro, que é o ângulo nessa articulação, Se a mão tem 7 dedos, cada um com 2 graus de liberdade na sua base, a macro `mao` deve ter $7 \times (1 + 2) = 21$ argumentos; e assim por diante.

Arquivos. Copie os arquivos da aula passada para uma nova sub-pasta 2020-10-28 da pasta `mc937` no seu computador. Edite o arquivo `main.pov`, conforme solicitado acima. Execute o comando `make` numa shell para gerar a imagem.

Originalidade. O arquivo de descrição `main.pov` deve ser construído manualmente, com um editor de texto comum, **sem** o auxílio de qualquer editor gráfico ou outra ferramenta de modelagem geométrica. Não é permitido copiar ou incluir quaisquer arquivos POV-Ray além dos fornecidos pelo professor ou escritos por você mesmo. Porém, é permitido re-usar arquivos ou trechos de código de exercícios anteriores.

Individualidade. Lembre-se de que todo trabalho prático é **individual**. Não é permitido pedir qualquer tipo de ajuda a colegas ou outras pessoas. Dúvidas devem ser tiradas apenas com o professor.

Exportação. Não se esqueça de **exportar seu arquivo main.pov até o final da aula para sua pasta WWW pública**

http://students.ic.unicamp.br/~raSEU_RA/mc937-2020-2/2020-10-28/

Comandos. Os seguintes comandos de POV-Ray são relevantes para esta tarefa:

Vetores canônicos. As variáveis x , y , e z são reservadas na linguagem POV-Ray, e tem valores $\langle 1,0,0 \rangle$, $\langle 0,1,0 \rangle$, e $\langle 0,0,1 \rangle$, respectivamente. Assim, por exemplo, `#local pp = 30*z - 5*x;` equivale a `#local pp = $\langle -5,0,30 \rangle$;`.

Rotação. Ao definir ou instanciar um objeto, o modificador `rotate` *vetor* pode ser usado antes do `}` final (como `translate`) para rodar o mesmo. Se o *vetor* é $\langle 20,10,60 \rangle$, o objeto será rodado por 20 graus em torno do eixo X , 10 graus em torno do eixo Y , e 60 graus em torno do eixo Z , nessa ordem. O sentido da rotação é sempre o contrário do que você espera. Por exemplo, `cylinder{ $\langle 0,0,10 \rangle$, $\langle 0,0,20 \rangle$, 5 rotate 20*x }` é um cilindro inicialmente vertical, rodado 20 graus em torno do eixo X (ou seja, inclinado 20 graus na direção do eixo Y).

Escala. Outro modificador que pode ser usado na mesma posição é `scale` *vetor*. Se o *vetor* é $\langle 2.0, 0.5, 1.0 \rangle$, o objeto será esticado por um fator de 2 na direção do eixo X e encolhido pela metade na direção do eixo Y , sem alteração na direção Z . Por exemplo, `box{ $\langle 4,4,4 \rangle$, $\langle 10,10,10 \rangle$ scale $\langle 2.0, 0.5, 1.0 \rangle$ }` equivale a `box{ $\langle 8,2,4 \rangle$, $\langle 20,5,10 \rangle$ }` Fatores de escala negativos invertem o objeto na direção correspondente; por exemplo, se `object{ luva }` gera uma luva para mão esquerda perto de $\langle 10,20,30 \rangle$, então `object{ luva scale $\langle -1.0, +1.0, +1.0 \rangle$ }` gera uma luva para a mão direita, perto de $\langle -10,20,30 \rangle$.

Combinando modificadores. É possível aplicar dois ou mais modificadores `translate`, `rotate`, e `scale` no mesmo objeto; porém, a ordem geralmente é importante. Dado `#local poste = cylinder{ $\langle 0,0,0 \rangle$, $\langle 0,0,10 \rangle$, 3 }`, as chamadas

```
object{ poste translate 10*z rotate 90*x scale  $\langle 2,3,4 \rangle$  }
object{ poste rotate 90*x translate 10*z scale  $\langle 2,3,4 \rangle$  }
object{ poste rotate 90*x scale  $\langle 2,3,4 \rangle$  translate 10*z }
```

etc. produzem resultados bem diferentes, assim como

```
object{ poste rotate 90*x rotate 90*z }
object{ poste rotate 90*z rotate 90*x }
```

Centro da rotação e escala. Note que os modificadores `rotate` e `scale` são sempre relativos à origem, e não ao centro do objeto. Para rodar um objeto *coisa* em relação a um ponto arbitrário `ctr`, por exemplo seu centro, o objeto deve ser deslocado de modo a colocar esse ponto na origem, antes de aplicar a rotação, e depois deslocado de volta: `object{ coisa translate -ctr rotate 30*x translate +ctr }`.

Lembrete. As palavras x , y , z , u , v , t são reservadas pela linguagem POV-Ray e não podem ser usadas como nomes de variáveis. Use letras maiúsculas, ou nomes com 2 ou mais letras.