MC937A/MO603A – Computação Gráfica - 2020-S2 - Jorge Stolfi
Trabalho de laboratório 06 - 2020-11-11
Robogarçom III - Animado

Objetivos: treinar a técnica básica de animação 3D por computador.

Enunciado. Ver para crer, diz o ditado. Só podemos acreditar que um robogarçom é capaz de movientos suaves e precisos se houver um vídeo mostrando isso. Taí a tarefa de hoje: produzie um vídeo com seu robogarçom executando um movimento repetitivo, envolvento várias partes do corpo.

Parte 1. Antes de começar a programar, **desenhe um esboço do seu robogarçom em pelo menos quatro poses ao longo do movimento planejado,** e apresente-o ao professor via Meet, quando solicitado, no início da aula.

Parte 2. Produza uma macro `quadro(tf)` que devolve o robô posição em um instante arbitrário `tf` do filme; sendo `tf=0` e `tf=1` produzem a posição inicial, e valores intermedários de `tf` produzem outras posições intermediárias, com variação suave.
Você vai precisar de uma macro `robo_interpola(tt, P0, P1)`, semelhante a `robo_mov` da aula passada que recebe dois vetores de parâmetros `P0` e `P1`, interpola um vetor `P` entre os mesmos, e chama `robo_vet` com esse vetor.

Arquivos. Copie os arquivos da aula passada para uma nova sub-pasta `2020-11-11` da pasta `mc937` no seu computador. `Baixe o novo arquivo Makefile e MOVIE.make de http://www.ic.unicamp.br/~stolfi/cursos/MC937-2020-2-A/progs/hand-out/2020-11-11 com wget ou com a função "Download"do seu browser.` Edite o arquivo `main.pov`, conforme solicitado acima. Execute o comando `make` numa shell para gerar a imagem.

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/~ra`*SEU_RA*`/mc937-2020-2/2020-11-11/`

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

Movimento contínuo por partes. Um movimento cíclico suave não pode ser executado com uma única interpolação de parâmetros `interpola3` (muito menos `interpola1`). Então é necessário usar várias interpolações, como segue:

```
#macro quadro(tf)
  #local T0 = 0.00;
  #local P0 = ...
  #local T1 = 0.25;
  #local P1 = ...
  #local T2 = 0.50;
  #local P2 = ...
  #local T3 = 0.80;
  #local P3 = ...
  #local T4 = 1.00;
  #local P4 = P0
  #if ((T0 <= tf) & (tf <= T1))
    #local tt = (tf - T0)/(T1 - T0);
    #local qd = robo_interpola(tt, P0, P1);
  #elseif ((T1 <= tf) & (tf <= T2))
    #local tt = (tf - T1)/(T2 - T1);
    #local qd = robo_interpola(tt, P1, P2);
  #elseif ((T2 <= tf) & (tf <= T3))
    #local tt = (tf - T2)/(T3 - T2);
    #local qd = robo_interpola(tt, P2, P3);
  #elseif ((T3 <= tf) & (tf <= T4))
    #local tt = (tf - T3)/(T4 - T3);
    #local qd = robo_interpola(tt, P3, P4);
  #end
  qd
#end
```

Por enquanto, sugiro usar interpolação afim (`interpola1`) na macro `robo_interpola(tt, P0, P1)`. Em aula futura veremos como usar (`interpola3`) para obter um movimento mais suave.

Chaamando a macro `quadro`. Na parte principal do `main.pov`, chame `object{ quadro(clock) }`.

Executando o `make`. Numa shell, execute `make fast`, `make movie`, ou `make CLOCK=0.1234 still` para gerar apenas um quadro.