COMPUTAÇÃO GRÁFICA

Prática 8 – Animação de Objetos

Ivan Nunes da Silva

Animação de Objetos

- Objetivos da Aula:
  - Apresentar as principais formas utilizadas no MATLAB para realização de animação de objetos.
  - Descrever os comandos computacionais envolvidos com a animação de objetos no MATLAB.
  - Realizar exercícios aplicativos envolvendo a animação de objetos.

2

Funcionamento do Comando “Plot3”

- Como visto anteriormente, o comando **plot** dos gráficos bidimensionais foi estendido para os tridimensionais com o **plot3**.
- O formato é o mesmo utilizado para os gráficos bidimensionais, exceto pelo fato de os dados estarem em conjuntos de três, em vez de aos pares.
- O formato generalizado de plot3d é o seguinte:
  - **plot3(x_1,y_1,z_1,s_1,x_2,y_2,z_2,s_2,x_3,y_3,z_3,s_3, ...)**
    - (x_n,y_n,z_n) são conjuntos de dados e s_n são strings de caracteres opcionais especificando cor, símbolos marcadores e/ou estilos de linha.

3

Funcionamento do Comando “Set”

- A instrução **set** permite alterar os atributos da estrutura do objeto **plot3** através de sua variável de manipulação.
- Sintaxe da instrução **set**:
  - **set(var_objeto, ‘atributo1’, valor1, ‘atributo2’, valor2, ..., ’atributo_n’, valor_n);**
  - **Exemplo:**
    - objeto = plot3(... conjunto de atributos ...)
    - set(objeto, ‘XData’, x, ‘YData’, y, ZData, z, ‘LineWidth’, 5, ‘LineStyle’, ‘:’, ‘Marker’, ‘o’, ‘MarkerEdgeColor’, ‘c’, ‘MarkerFaceColor’, ‘r’)
      - ‘LineWidth’ ➔ Espessura de linha em pontos {valor default é 0.5 pontos}.
      - ‘LineStyle’ ➔ Estilo de linha {-’, ‘--’, ’:’, ‘-.’, ’none’}.
      - ‘Marker’ ➔ Marcador de pontos {‘+’, ’o’, ’*’, ’.’, ’x’, ’s’, ’d’, ’none’}.
      - ‘MarkerSize’ ➔ Tamanho do marcador em pontos {valor default é 6 pontos}.
      - ‘MarkerEdgeColor’ ➔ Cor da linha do marcador {‘y’, ’m’, ’c’, ’r’, ’g’, ’b’, ’w’, ’k’}.
      - ‘MarkerFaceColor’ ➔ Cor da face do marcador {‘y’, ’m’, ’c’, ’r’, ’g’, ’b’, ’w’, ’k’}.

4

Formas de Animação no MATLAB

- O Matlab fornece duas formas para se realizar animação de objetos, ou sejam:

1. **Quadro Sobreposto** → apaga e então desenha continuamente os objetos na tela, realizando mudanças incrementais em cada redesenho.
2. **Quadro Subsequente** → realiza a captura quadro a quadro do movimento do objeto, reproduzindo posteriormente em sequência.

5

Funções Para "Quadro Sobreposto"

- **drawnow** → função que força o matlab a descarregar os objetos gráficos pendentes para a janela ativa.
  - Exemplo: var_objeto = plot3(x(i), y(i),t(i));
    set(var_objeto, 'xdata',x(i), 'ydata', y(i), 'zdata', t(i));
    drawnow; **{ Força o matlab a traçar o gráfico }**
- **erasemode** → atributo do objeto gráfico que informa como o matlab alterará os objetos (imagens) que serão enviados para a janela gráfica. Os principais valores assumidos são:
  - 'normal' → significa que a janela gráfica será limpa antes.
  - 'none' → significa que a janela gráfica não será limpa.
  - 'xor' → traça e apaga a imagem executando uma operação de ou-exclusivo com as cores da tela antes e depois.
  - Exemplo: var_objeto = plot3(x(i), y(i),t(i));
    set(var_objeto, 'erasemode', 'none'); **{ não limpa o gráfico }**

6

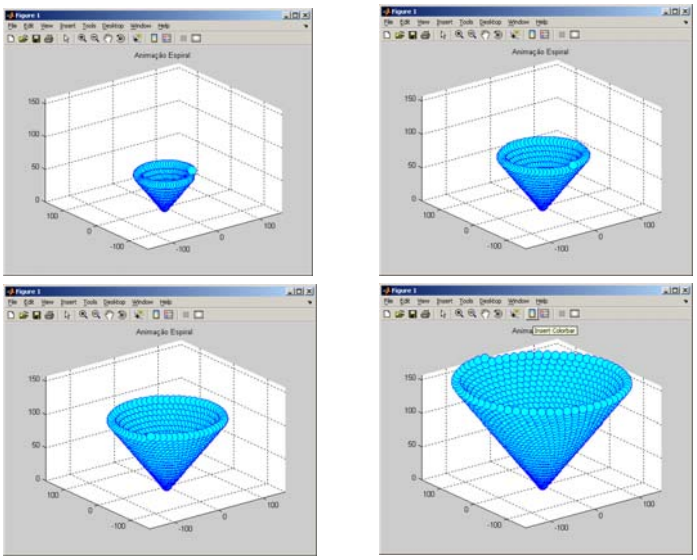
Funções Para “Quadro Sobreposto”

- **Exercício 1:** Uma partícula descreve a trajetória em movimento circular uniforme ao redor da envoltória de um cone espiralado. Realize a animação de tal movimento da seguinte forma:
  - A partícula (ponto) será representada por um marcador em “círculo”, tendo cor de face ciano e tamanho 12.
  - A envoltória espiralada será composta de 25 revoluções com passo discreto de 0.1, tendo a mesma a seguinte equação paramétrica:
    - t = 0:0.1:25*(2*pi);
    - x = t.*sin(t);
    - y = t.*cos(t);
  - Realize a animação sem apagar os movimentos anteriores.
  - Coloque uma pausa de 0.1 segundos entre os traçados.

7

Funções Para “Quadro Sobreposto”

- Resultado da animação do Exercício 1.



8

Funções Para "Quadro Subsequente"

- **getframe** → função que captura um quadro (fotografia) da figura que está sendo exibida na janela atual.
  - Exemplo: *M*(*i*) = getframe; {Armazena o quadro atual na *i*-ésima posição de *M*}.
    Exemplo: *M*(*i*) = getframe(gcf); {Captura apenas o gráfico da figura}.
- **movie(*M, n, FPS*)** → movimenta em sequência todos os quadros armazenados em *M* durante *n* vezes. O parâmetro *FPS* representa a velocidade em quadros/seg. Se omitido, então assumirá 12 quadros/seg.
  - Exemplo: movie(*M*, [*number_repeats* *frame_order*]);

9

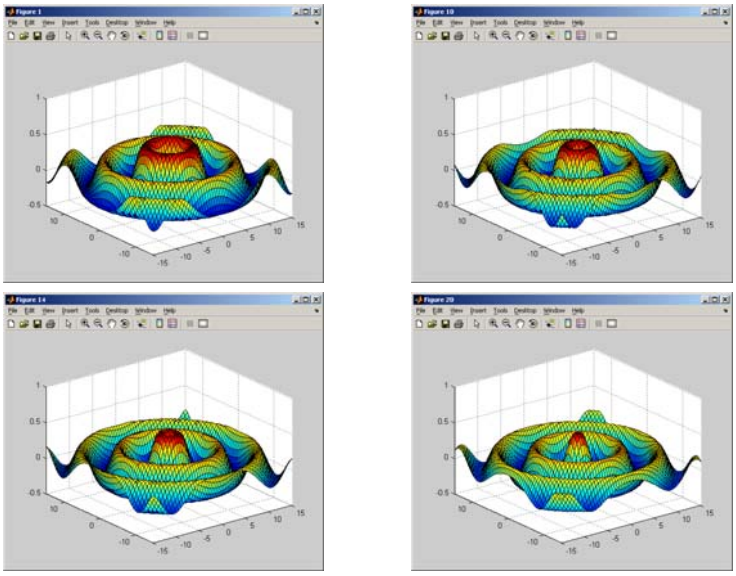
Funções Para "Quadro Subsequente"

- **Exemplo 2** → Animar o comportamento da função de Bessel conforme mostrado a seguir:
  - Gerar 20 quadros (j=1..20) da função de Bessel seguinte, sendo que cada quadro é dado por:
    - z = besselj(3, (j-1)*0.2 + sqrt(x.^2 +y.^2))
    - O grid é definido entre –15 e 15, tendo espaçamento de 0.5.
    - Eixo X → -15 a 15 ; Eixo Y → -15 a 15 ; Eixo Z → -0.5 a 1.0
  - Após a geração dos quadros, confeccionar a animação em 39 quadros na seguinte ordem:
    - Em sequência crescente → Varrendo de 1 até 20.
    - Em sequência decrescente → Varrendo de 19 até 1.
    - Repetir a animação na sequência acima durante 10 vezes.
    - Assumir velocidade de transição default de 12 frames/seg.
  - Utilizar o comando Figure('Renderer', 'zbuffer') no início caso sua máquina tenha pouca memória.

10

Funções Para "Quadro Subsequente"

- Resultado da animação do Exercício 2.



11

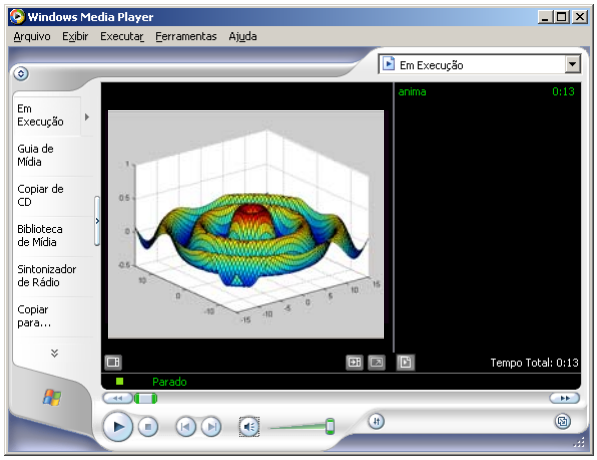
Funções Para Geração de Vídeo

- **VideoWriter**(*'nome_arquivo.avi'*) → gera arquivo ".avi" para salvar a animação em vídeo.
  - Exemplo: *mov* = VideoWriter(*'anima.avi'*); onde *mov* é a *variável-arquivo*.
- **open**(*variável_arquivo*) → abre o arquivo para leitura ou escrita de frames.
  - Exemplo: open(*mov*);
- **writeVideo**(*variável_arquivo*, *variável_frame*) → adiciona sequencialmente um frame ao arquivo que conterá a animação.
  - Exemplo: writeVideo(*mov*, *M*);
- **Close**(*variável_arquivo*) → fecha o arquivo de animação.
  - Exemplo: close(*mov*);
- **VideoReader**(*'nome_arquivo.avi'*) → lê um arquivo de animação de vídeo para a workspace do matlab.
  - Exemplo: *mov* = VideoReader(*'nome_arquivo.avi'*)

12

Funções Para Geração de Vídeo

- **Exemplo 3** → Gerar o vídeo referente ao exemplo do exercício 2.
  - Utilize cinco repetições.



13