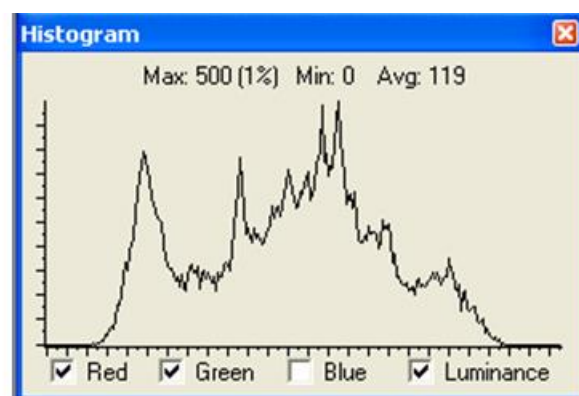
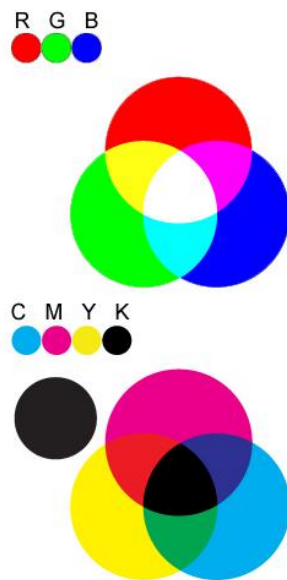


Introdução

1. Overview do que será visto no curso (e programado)

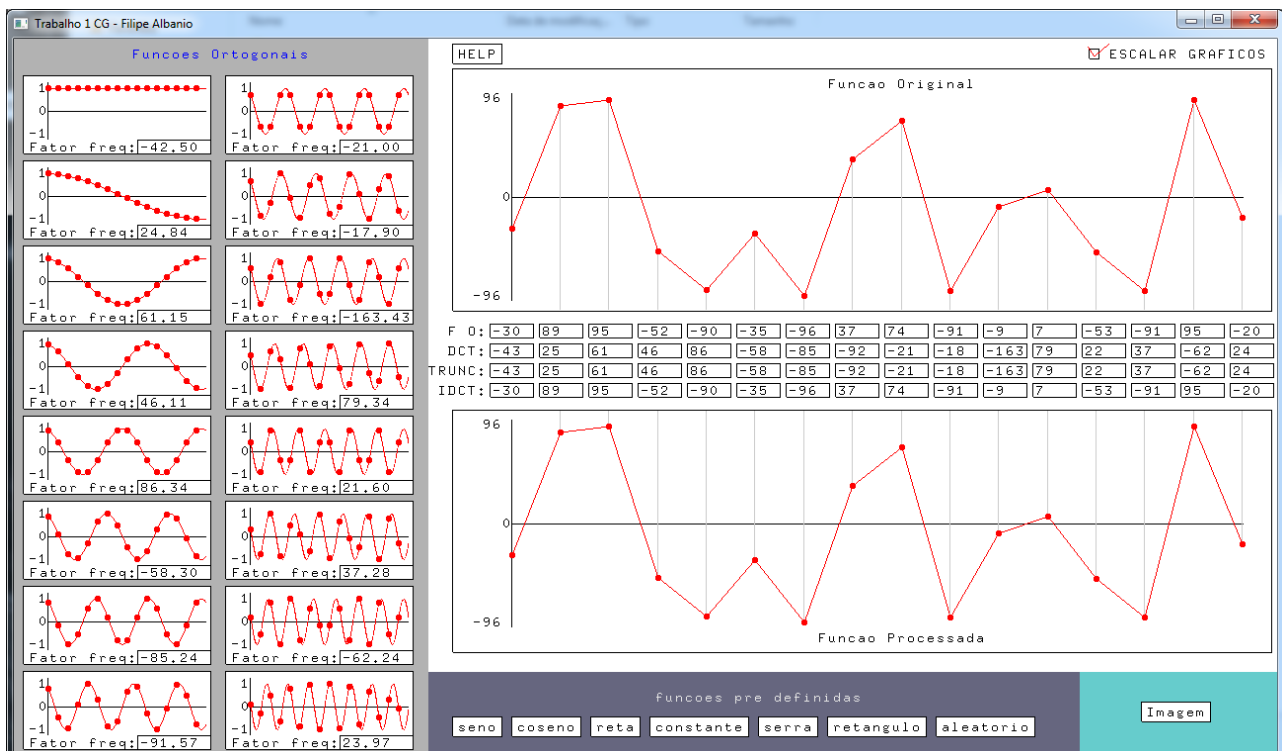
Fundamentos de Imagem e Cor



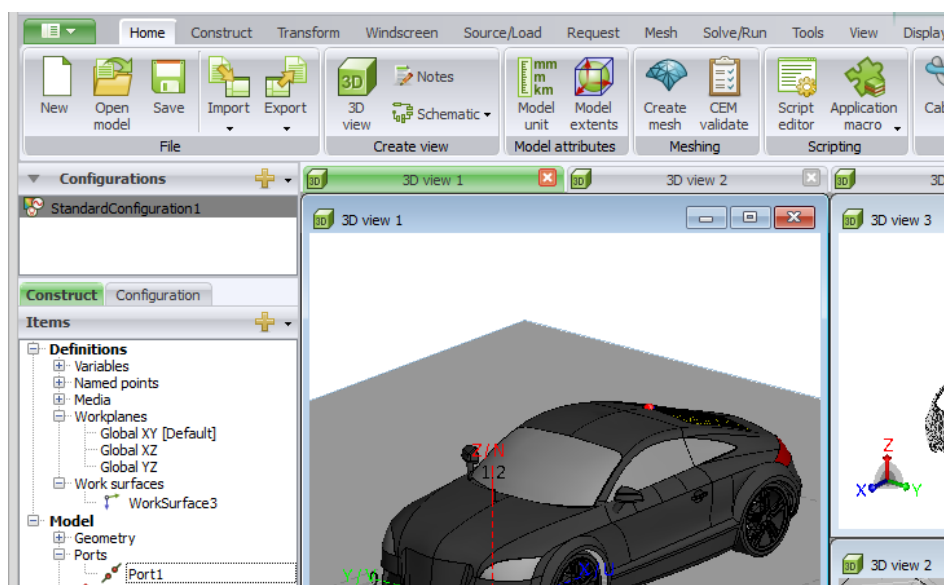
- 1.5.2 – Dispositivos de saída.
 1.6 – Fundamentos de Cor e Imagem.
 1.6.1 – Sistemas RGB, HSV, CMYK, qua
 1.6.2 – Formatos de Imagens: BMP, JPG.

UNIDADE 2 – FUNDAMENTOS MATEMÁTICOS

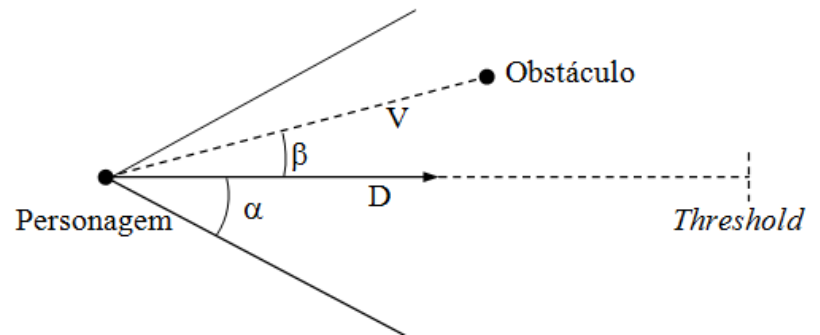
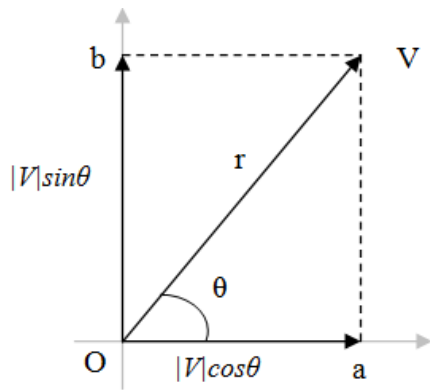
- 2.1 – Álgebra Linear e Computação Gráfica
 2.2 – Sistemas de Coordenadas: Cartesiano



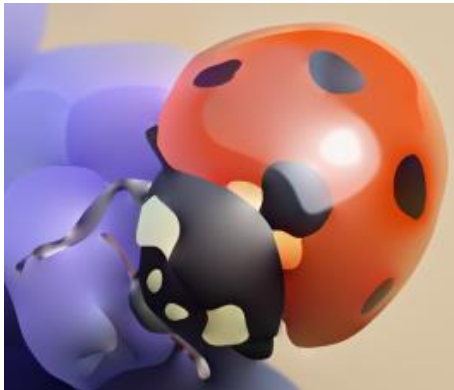
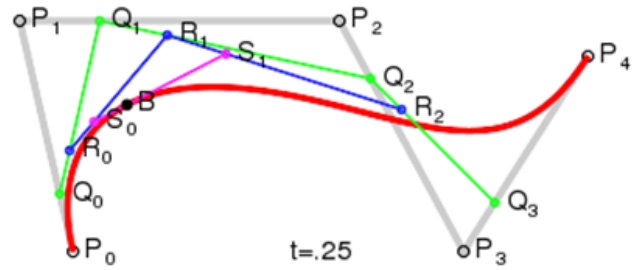
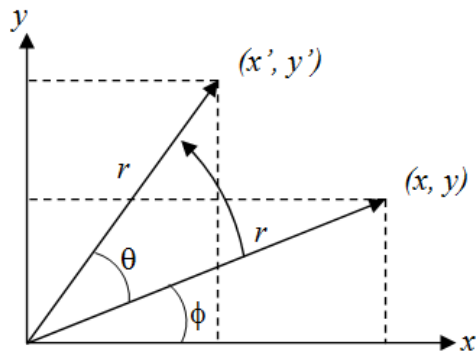
Interfaces gráficas

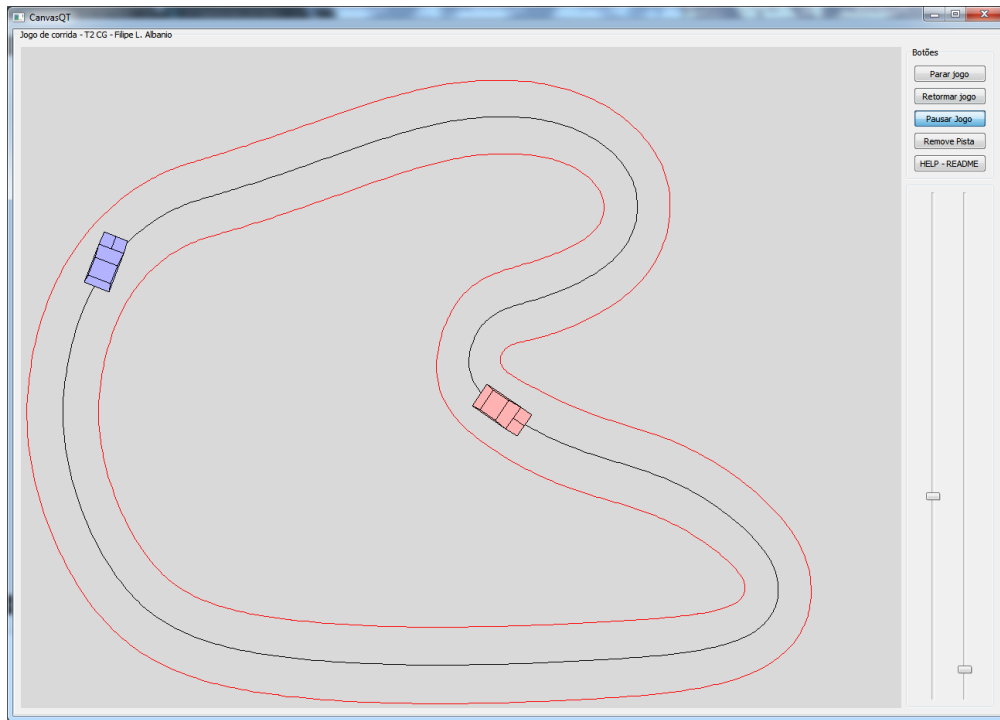


Base matemática

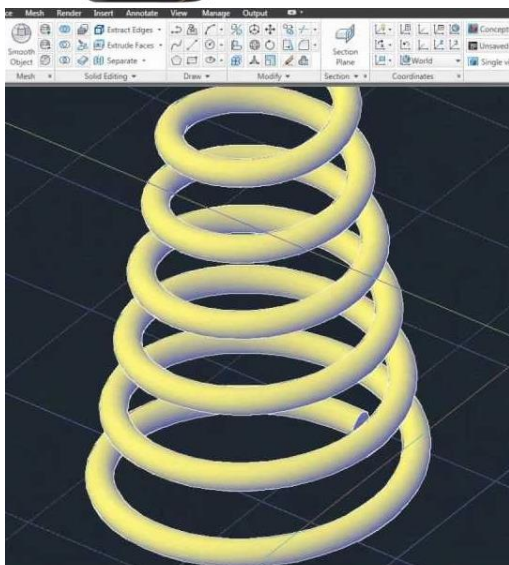
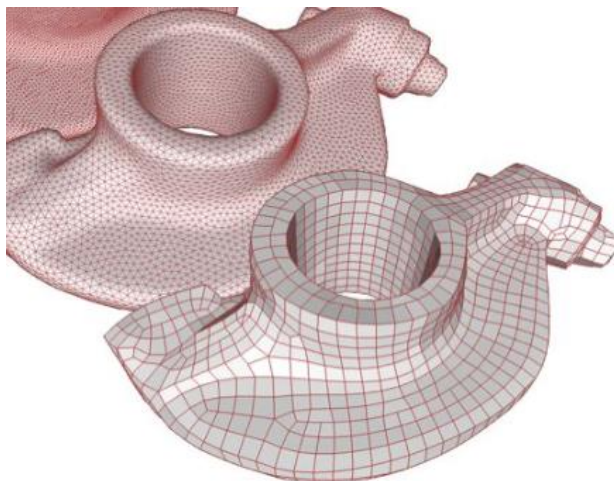


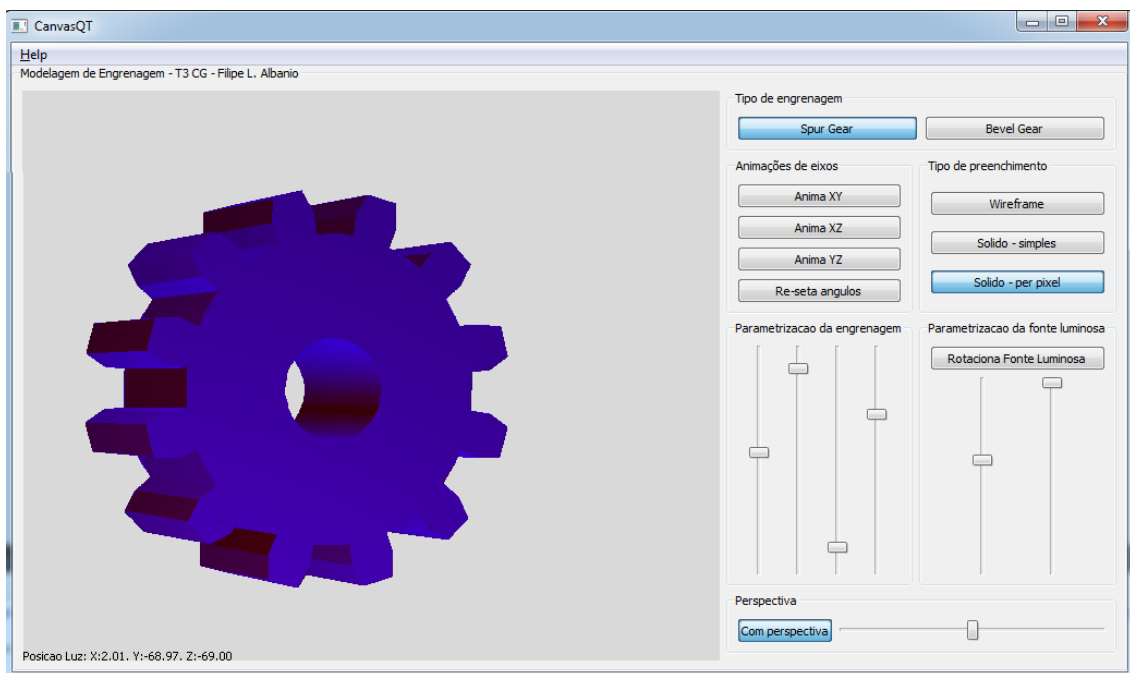
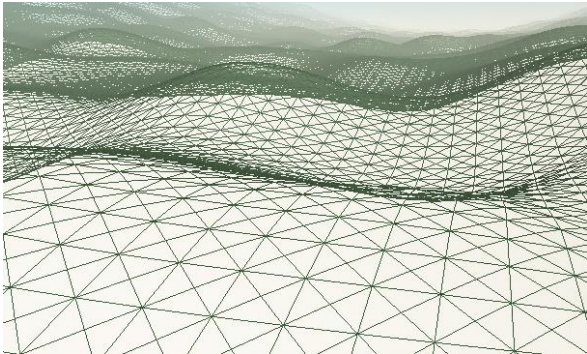
Computação Gráfica 2D





Computação Gráfica 3D





Síntese de Imagens





2. Visão geral dos Sistemas Gráficos

- CAD (*Computer-Aided Design*): projetos de engenharia para modelagem e visualização de peças, simulações de ferramentas e máquinas, plantas de casas, maquetes de edifícios;
- Apresentação de dados: gráficos, estatísticas;
- Arte computacional: Logomarcas, trabalhos artísticos, propaganda, efeitos especiais;
- Entretenimento: filmes, jogos eletrônicos;
- Educação e treinamento: cabines de simulação de voo, etc;
- Visualização: visualização científica, funções matemáticas;
- Processamento de Imagens.

Em todas estas áreas, a CG permite visualizar objetos que:

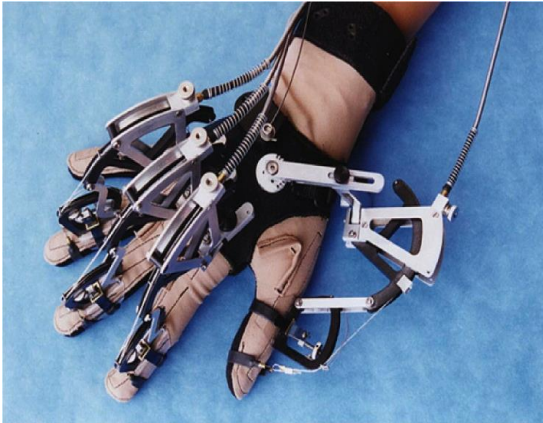
- ainda em construção
- estão fora do alcance da percepção visual
- Fogem da nossa realidade tridimensional

3. Dispositivos gráficos

■ Entrada

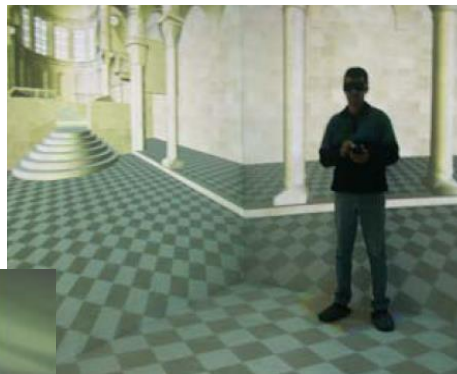
- Mouse 2D/3D
- Teclado
- Joystick
- Spaceball – 6 graus de liberdade: 3 de rotação e 3 de translação (**Six degrees of freedom (6DOF)**) (<http://www.3dconnexion.com/products/3a2.php>)
- Luva
- Mesa e caneta digitalizadora (2/3D)
- Scanner
- Tela sensível ao toque
-





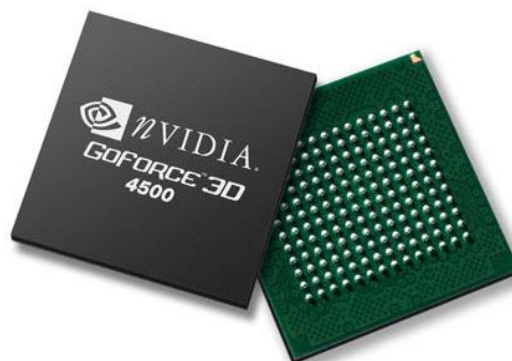
■ Saída

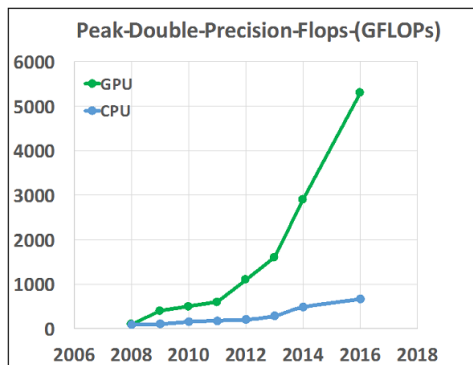
- Monitor + placa de vídeo: CRT, LCD
- Impressora: laser, jato de tinta, BW/cor
- Óculos (Relacionado principalmente com Realidade Virtual)
- Canhão



4. Processamento gráfico

- Software (CPU) + hardware (GPU)





Fabricantes

- **Workstations (passado)**
 - Silicon Graphics (<http://www.sgi.com/>)
 - Sun (<http://www.sun.com/>)
- **PCs**
 - Nvidia (<http://www.nvidia.com/page/home.html>)
 - ATI (<http://www.ati.com/>)
 - Intel

5. Evolução da Área

- **Hardware**
 - Monitores
 - Placas gráficas
 - Outros dispositivos
- **Técnicas**
 - 1978 - Mapeamento de texturas (Blinn).
 - 1980 - Primeiro *Ray-tracing* (Whitted).
 - 1992 - Surge a linguagem de programação OpenGL
- **Aplicações**
 - Realidade Virtual
 - Jogos, interatividade
 - Realismo
 - Filmes

6. Interfaces Gráficas

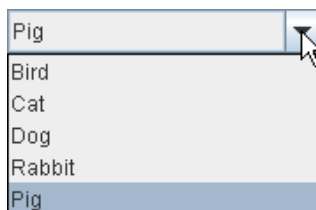
Além de permitirem a exibição de informação gráfica como imagens, animações, etc, um dos principais objetivos de interfaces gráficas é auxiliar o usuário na **interação** com a aplicação. Surgiram para abolir a utilização de consoles textuais e passagem de argumentos por linha de comando. Temos como exemplos a interface gráfica do Windows.

Uma interface gráfica pode disponibilizar ao programador uma simples canvas (área de desenho), como no caso da Canvas 2D (<http://www-usr.inf.ufsm.br/~pozzer/>), bem como também Widgets (componentes gráficos de interface) que permitem uma fácil manipulação de informação e eventos de interação com o usuário. Sob a canvas, o programador pode, se julgar necessário, desenvolver seus próprios widgets personalizados.

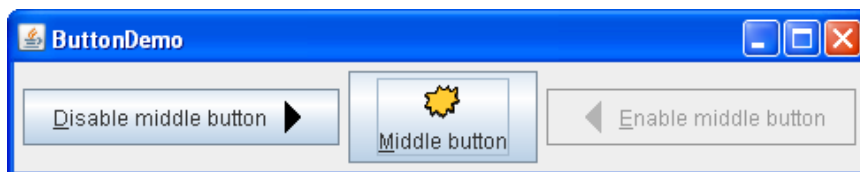
Ao se utilizar APIs gráficas, geralmente um grande número de widgets já estão disponibilizados. São exemplos de APIs para a linguagem C/C++ QT (http://www-usr.inf.ufsm.br/~avelar/tutorial_qt/), Fox, GTK, xwWidgets (<http://www.vconrado.com/wx/>), dentre outras. Para a linguagem Java, por exemplo, pode-se fazer uso do Toolkit Swing. O Swing foi desenvolvido para prover um conjunto mais sofisticado de componentes de interface (GUI – *Graphical User Interface*) do que o AWT (*Abstract Windows Toolkit*).

Nas seguintes figuras ilustra-se alguns exemplos de widgets do Toolkit Swing do Java (<http://72.5.124.55/docs/books/tutorial/uiswing/components/label.html>).

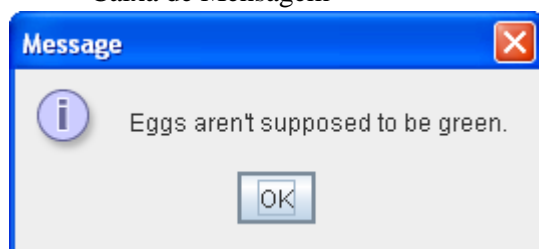
Combo Box



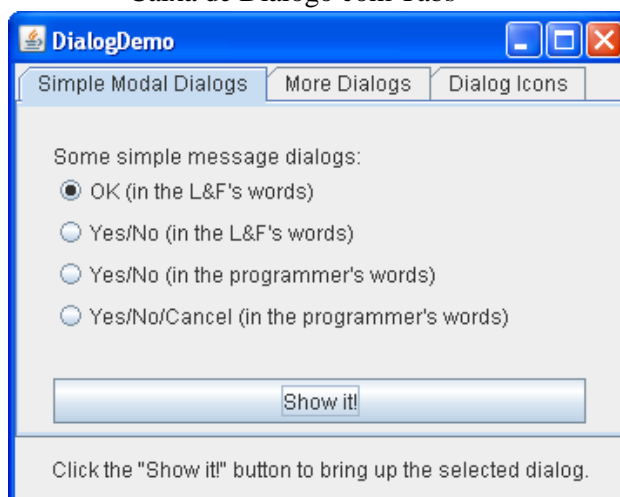
Botões



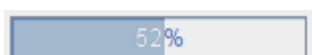
Caixa de Mensagem



Caixa de Diálogo com Tabs

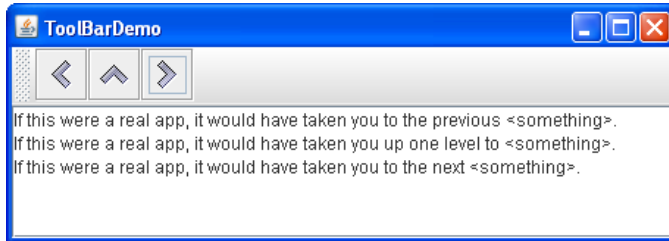


Barra de progresso

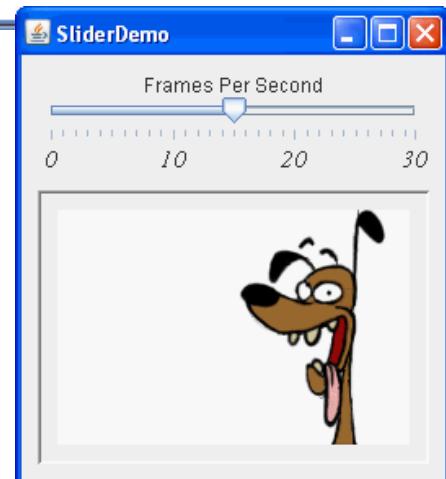
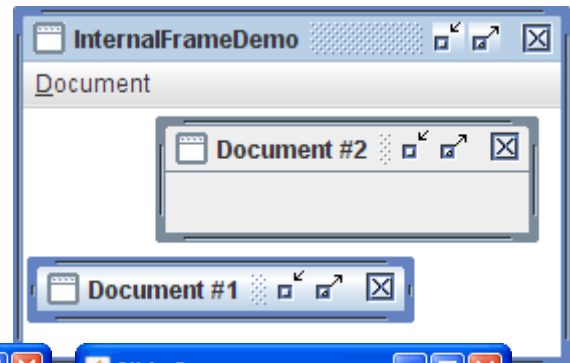
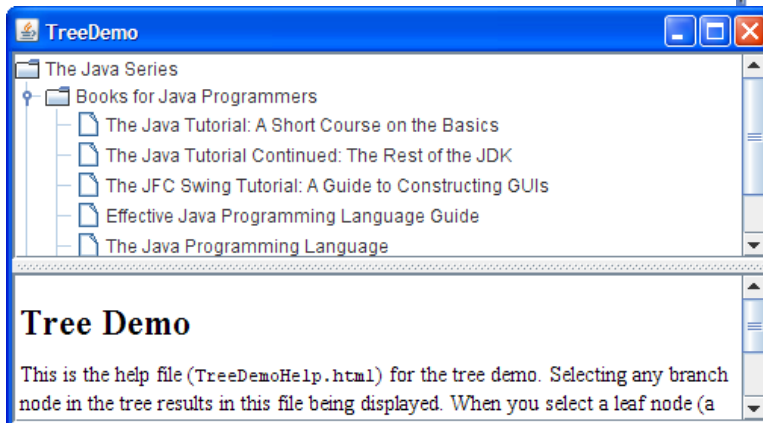


Frames

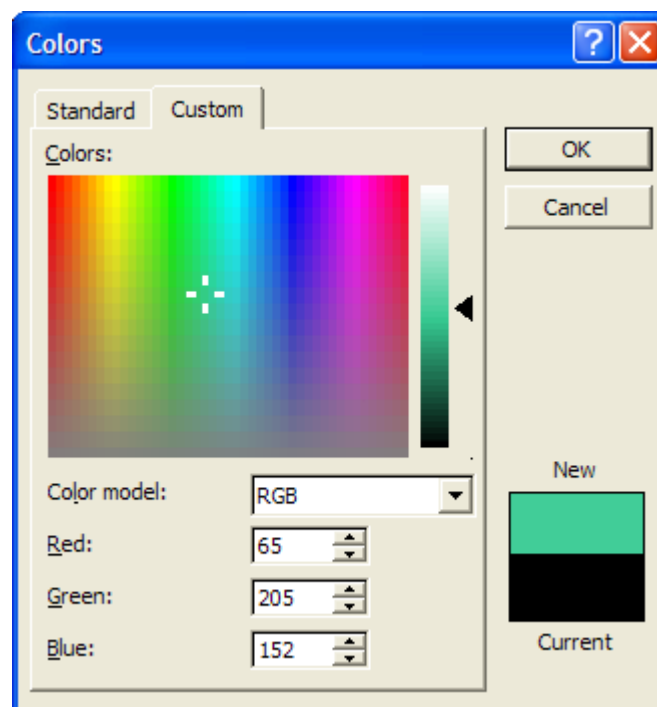
Barra de Ferramentas



Árvore

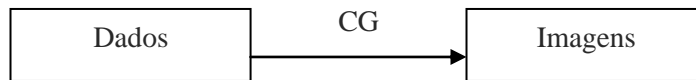


Combinando-se os diversos widgets, pode-se construir interfaces mais avançadas, como a usada para fazer a escolha de cores no modelo HSL, usada pelo programa Word da Microsoft.

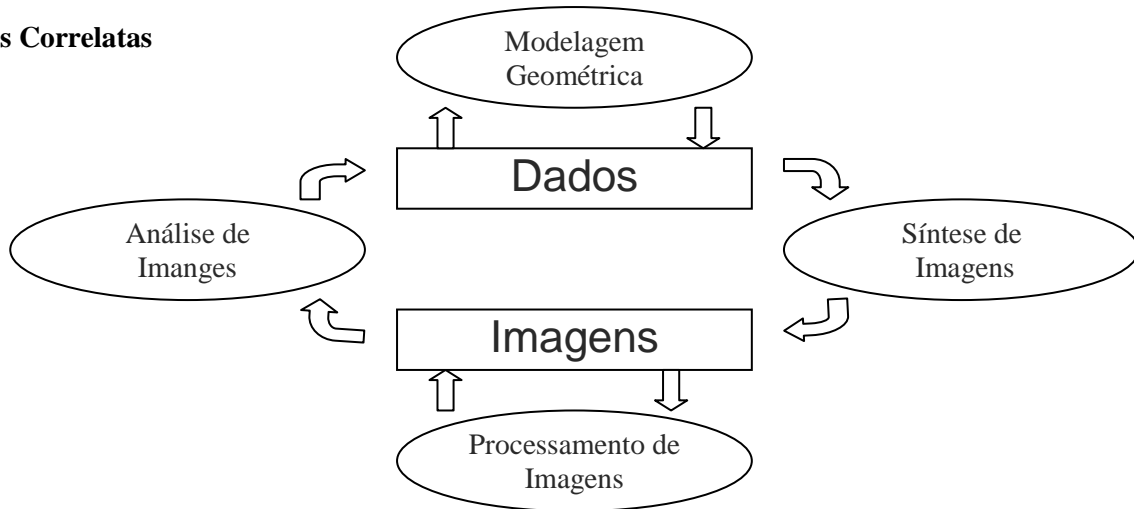


7. CG – Computação Gráfica

- Teve início na década de 50.
- Definição: “conjunto de métodos e técnicas para transformar dados em imagens através de um dispositivo gráfico”.



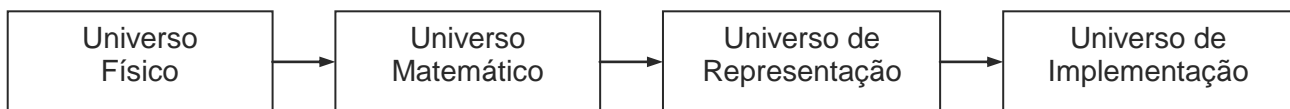
Áreas Correlatas



8. Paradigmas de Abstração

Na matemática aplicada que faz uso de métodos computacionais, como em CG, um paradigma de abstração (quatro universos) que se aplica em geral consiste em estabelecer quatro universos (conjuntos). Este paradigma se baseia no fato de que para estudar um determinado fenômeno ou objeto do mundo real no computador, associamos ao mesmo um modelo matemático, em seguida procuramos uma representação finita do modelo associado que seja passível de implementação [Velho].

- Universo Físico: objetos do mundo real
- Universo Matemático: descrição abstrata do mundo físico
- Universo de Representação: descrições simbólicas e finitas
- Universo de Implementação: uso de estruturas de dados para representar descrições do universo de representação.

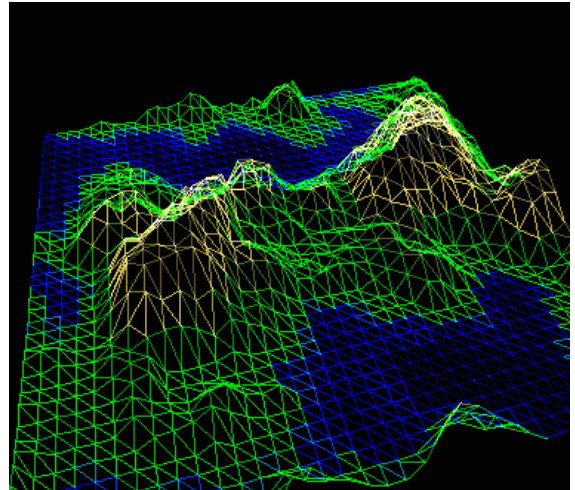


Exemplo: Representação de um terreno (Imagem)

- Pode ser definido por um mapa de alturas. Cada ponto tem uma elevação.
- No universo matemático, o mapa de alturas (*height field*) define uma função $F: U \subset \mathbb{R}^2 \rightarrow \mathbb{R}, z = f(x, y)$ onde (x, y) são as coordenadas do plano e z a altura corresponde.
- Geometricamente o terreno é descrito pelo gráfico da função de alturas f

$$G(f) = \{(x, y, f(x, y))\}$$

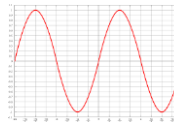
- Para **representar** o terreno, pode-se utilizar um reticulado de pontos (xi, yj) , obtido pelo produto cartesiano dos conjuntos Px e Py , onde
 $Px = \{x0 < x1 < \dots < xn\}$ e
 $Py = \{y0 < y1 < \dots < yn\}$
 $U = [a, b] \times [c, d], a \leq x \leq b$
- A representação do terreno é dada pela matriz de alturas $zij=f(xi, yj)$, onde (xi, yj) representa um vértice do reticulado.
- Esta representação é chamada representação por amostragem uniforme.
- Para **implementação**, pode-se utilizar uma matriz da linguagem C: `int matriz[lin][col]`.



9. Exercícios práticos utilizando a CanvasGlut (Disponível no site da disciplina)

Aquecimento:

1. Programa para desenho de um quadrado verde.
2. Programa para movimentação lateral de um quadrado com o uso das setas direcionais do teclado.
3. Programa para movimentação de um círculo com o uso do mouse.
4. Programa para plotar polinômios. Ex: $3x^3 - 10x^2 + 5x + 4$



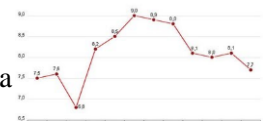
5. Programa para plotar as funções seno, cosseno e tangente.



6. Programa para desenho de gradientes de cores.

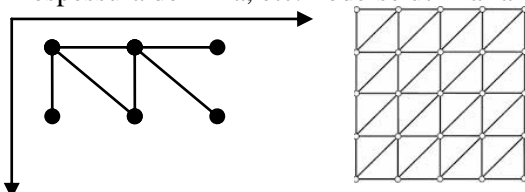


7. Programa para desenhar um alvo com círculos concêntricos.
8. Programa para entrada de dados (amostras) com o uso do mouse. Ao mover o cursor, a coordenada y do mouse deve ser armazenada num vetor indexado pela coordenada x do mouse.



Mais interessantes:

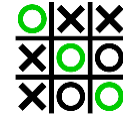
1. Faça um programa em C/C++ para gerar/ler um arquivo binário que armazena um reticulado de pontos. Cada ponto deve conter um valor que representa uma altura no intervalo $[0, 255]$. O arquivo deve ter um header que indica as dimensões do reticulado. Após, faça um programa para ler o arquivo gerado no exercício anterior e renderizar na tela, em 2D, a malha poligonal em wireframe, como mostrado na seguinte figura. O programa pode permitir escolha de cor, zoom, estilo de linha, espessura de linha, etc. Pode-se utilizar a valor de cada ponto para definir a cor de desenho da malha.



2. Implementar uma calculadora com um display digital. Os botões (dos números) devem ser pressionados com o uso do mouse. Os dígitos do display devem ser feitos com 7 linhas ou com 7 polígonos.

Jogos:

1. Implementar uma versão simplificada do jogo Pac-man sem inimigos (somente a movimentação do avatar do jogador)
2. Implementar o jogo Space Invaders sem uso de IA.
3. Implementar o jogo da velha sem IA: apenas o tabuleiro e inserção de jogadas alternadas entre O e X. Quando for feita uma



Interfaces gráficas:

1. Estude e desenvolva aplicativos simples com uso das seguintes GUIs: QT (<https://www.qt.io/>) e Dear ImGui (<https://github.com/ocornut/imgui>).