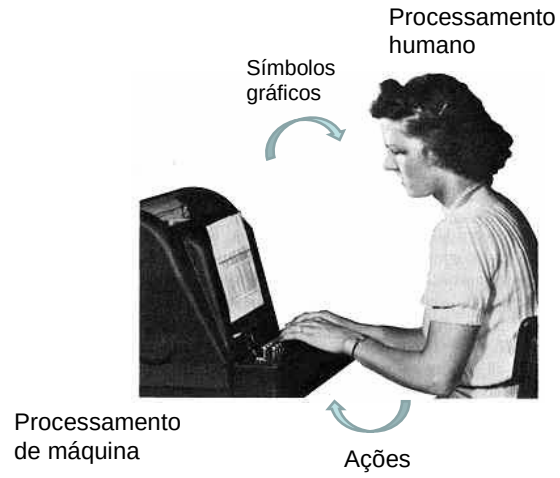
Sistemas Gráficos Interativos

Rogers & Adams: Capítulo 1
Redbook: Capítulo 1
Apostila: Capítulo 2

Interações

Computador suporte em processamentos

Interagir com o computador através de comunicação visual.



Arquitetura Vetorial

Controlador de "*Display*"

Dispositivos de Saída

Processamento Gráfico

Dispositivos de Entrada

Arquitetura *Raster*

Dispositivos de Saída

Controlador de Vídeo

Dispositivos de Entrada

Processamento Gráfico

Memória de Vídeo (*frame buffer*)

Dispositivos de Saída

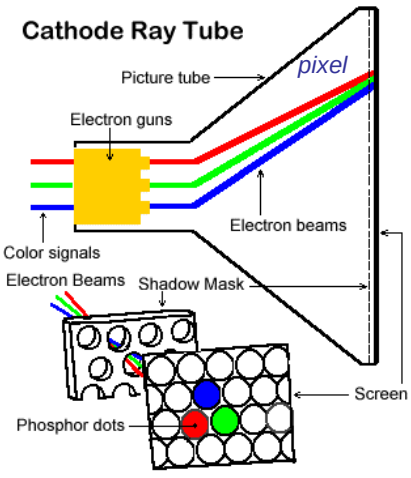
Periféricos capazes de transmitir os resultados de um processamento digital ao "mundo exterior".

Classificação

- Quanto ao modo de saída
  - visual
  - verbal
- Quanto à persistência
  - removível
  - permanente
- Quanto ao tipo de dados
  - discreto
  - contínuo

Dispositivos de Saída

Monitores CRT

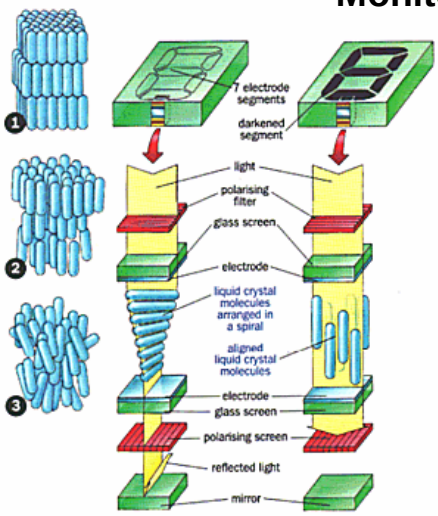


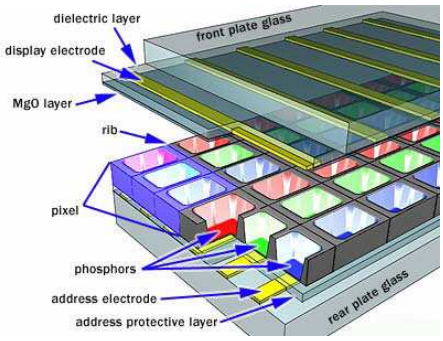
Tubos de Raios Catódicos (CRT)

Shadow Mask: fina folha metálica para evitar que feixe de elétrons atinja células vizinhas. Aumenta a nitidez da imagem.

Dispositivos de Saída

Monitores LCD



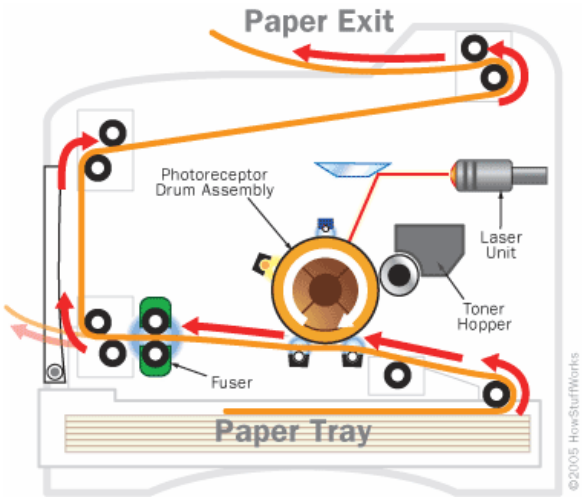


LCD – *Liquid cristal displays*

- Luz pode ser polarizada
- Cristal líquido pode transmitir e polarizar luz
- Estrutura do cristal líquido pode ser alterada pela corrente elétrica
- Há eletrodos transparentes

Dispositivos de Saída

Impressoras

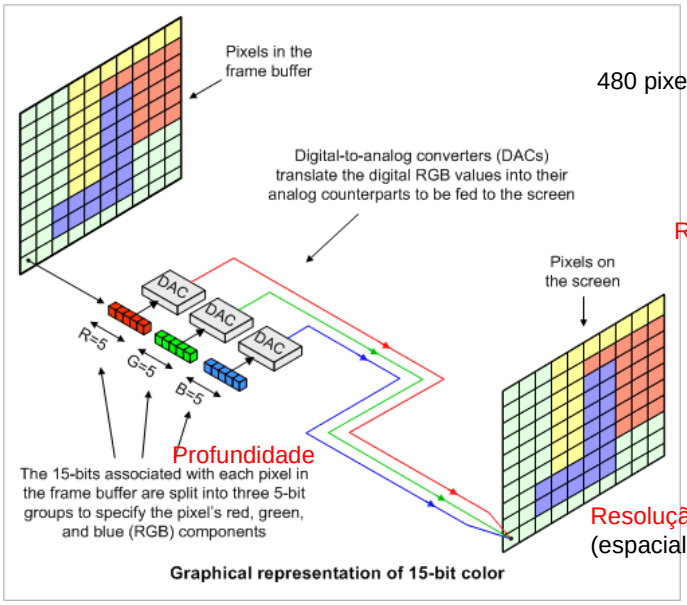


Princípio de funcionamento:
carga estática

Dispositivos Lógicos de Saída

Periféricos sob o ponto de vista de programação:
- resolução de saída (quantidade de *pixels*)
- "resolução" de cada *pixel* (tamanho de fragmento)
- quantidade de memórias de vídeo (*frame buffers*)
- modo de exibição: indexado e RGB

Memória de Vídeo



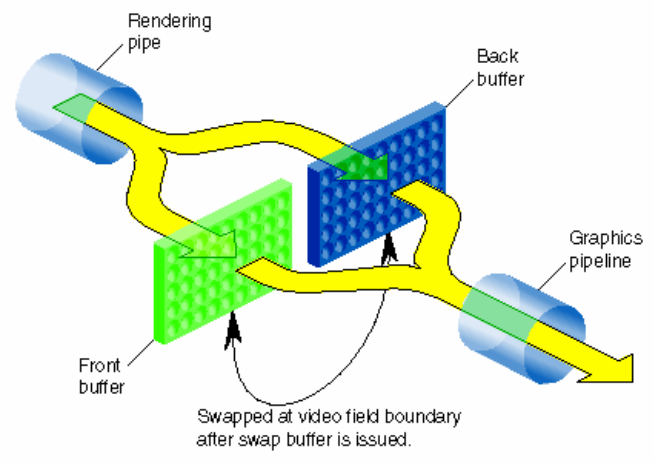
Viewport

480 pixels

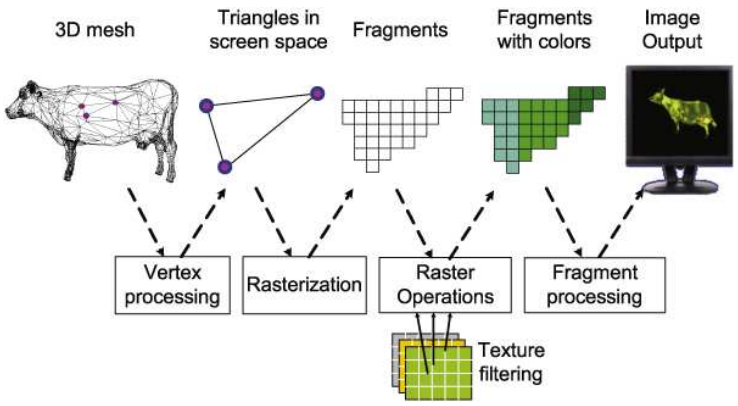
640 pixels

Razão de aspecto: 4:3

Memória de Exibição Dupla

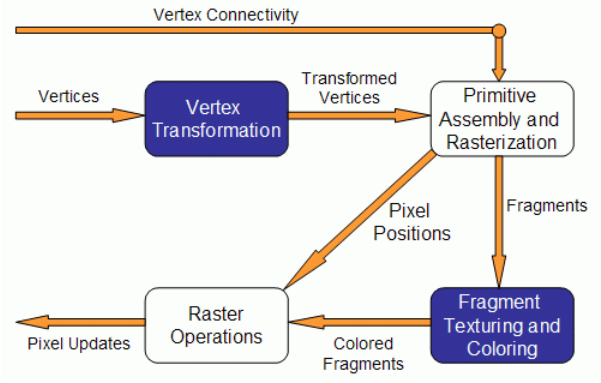


Processamento Gráfico *Raster*



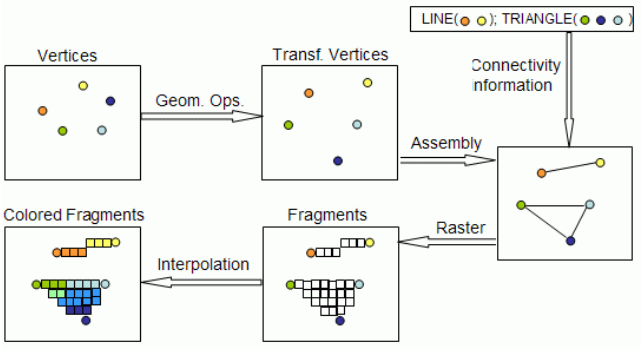
http://www.opengl.org/

OpenGL: Fluxo de Processamento



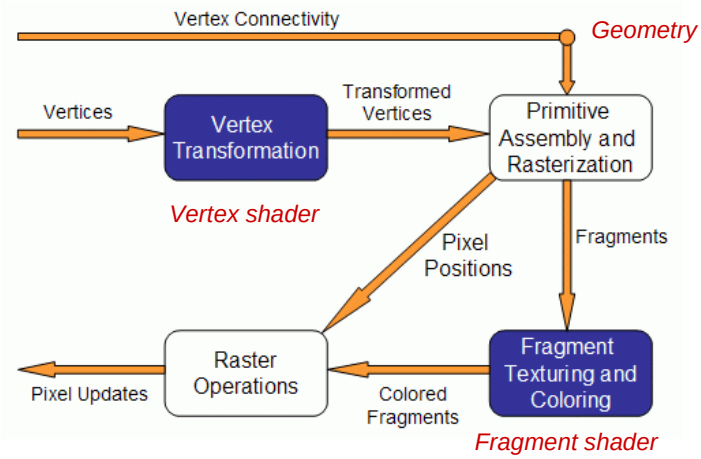
http://zach.in.tu-clausthal.de/teaching/cg_literatur/glsl_tutorial/index.html

OpenGL: Fluxo de Processamento

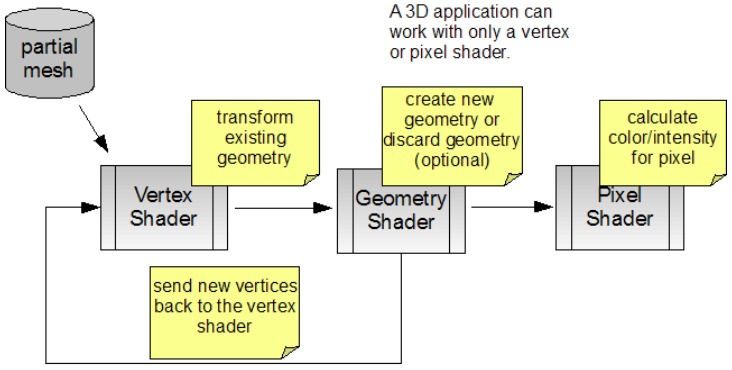


http://zach.in.tu-clausthal.de/teaching/cg_literatur/glsl_tutorial/index.html

OpenGL: Fluxo de Processamento Programável



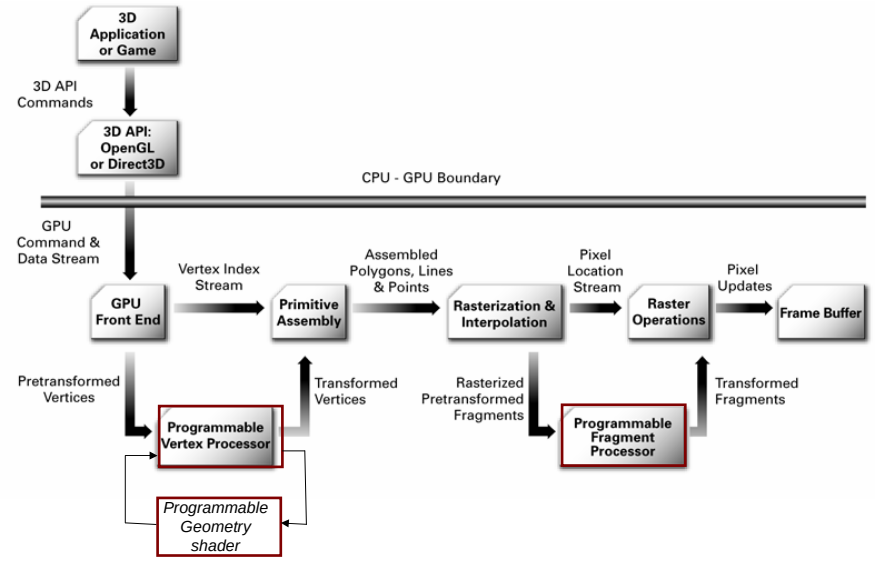
OpenGL – fluxo programável



http://www.khronos.org/files/**opengl-quick-reference**-card.pdf

http://www.cs.umd.edu/users/mount/427/OpenGL/ogl_ref/ogl_ref.html

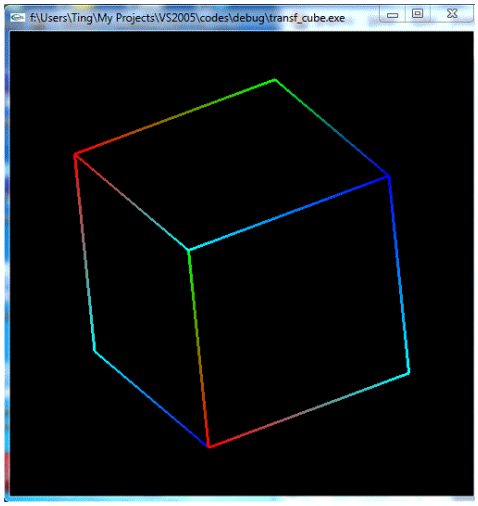
OpenGL – fluxo programável



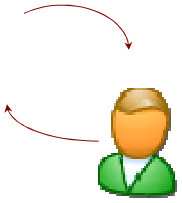
Instalação de OpenGL

http://nehe.gamedev.net/lesson.asp?index=01

Interações com Gráfico *Raster*

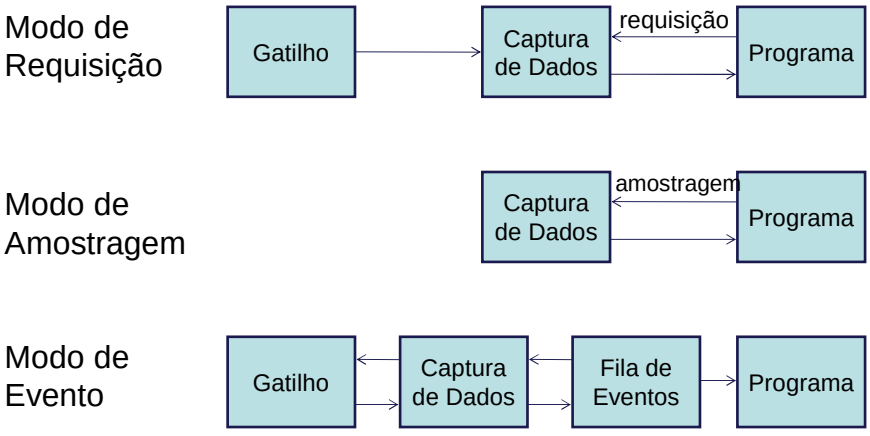


transf_cube



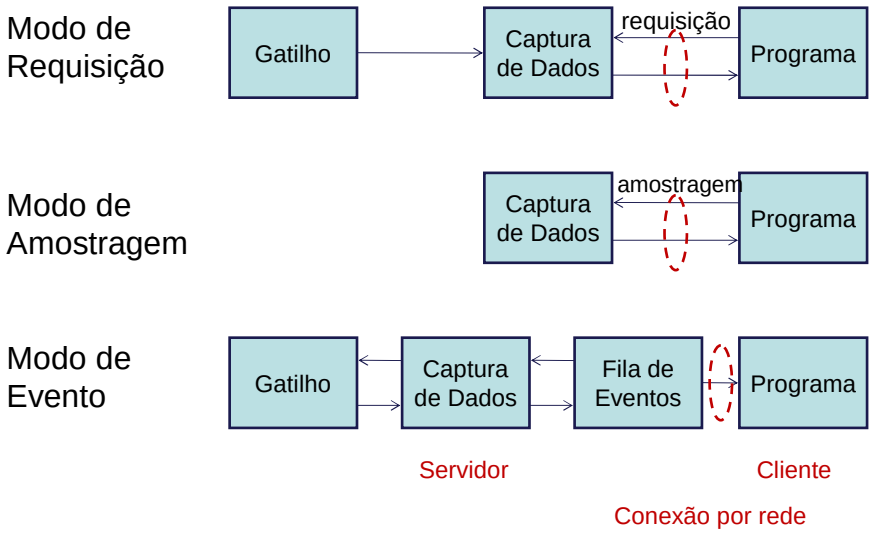
http://www.dca.fee.unicamp.br/courses/IA725/1s2006/program/samples.html

Modos de Entrada



Modos de Entrada

Arquitetura em Rede



Dispositivos de Entrada

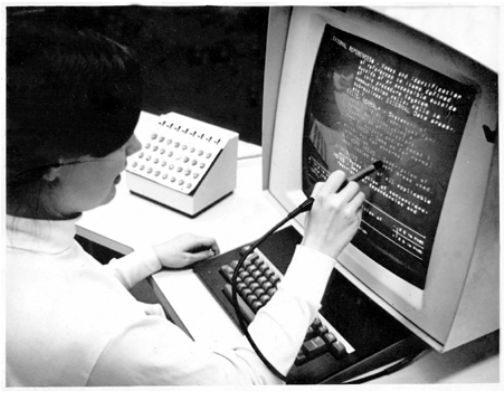
Periféricos capazes de captar dados gerados pelos usuários

Classificação

- Quanto ao modo de entrada
  - Visual (sinais luminosos)
  - Áudio (sinais sonoros)
  - Pressão (forças)
  - Movimento
- Quanto ao tipo de dados
  - Discreto
  - Contínuo
- Quanto ao grau de liberdade
  - 2-, 3-, 6-, ou mais

Dispositivos de Entrada
Caneta Óptica (*Lightpen)*

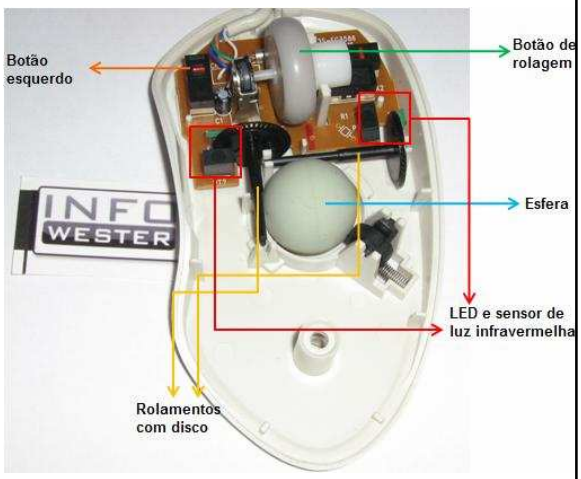
•Controle por sinal luminoso
•Discreto
•2 graus de liberdade
•Aquisitor de pontos/Localizador (posição absoluta)/Selecionador/ Identificador



Não pode ser utilizado em telas LCD!

Dispositivos de Entrada
Mouse e Trackball

•Controle por movimento
•Contínuo
•2 graus de liberdade
•Localizador (posição relativa)/ Selecionador/ Identificador



Dispositivos de Entrada
Teclado

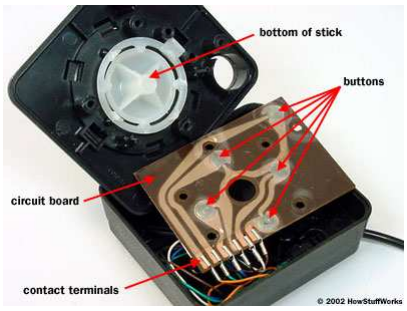
•Controle por movimento (de dedo)
•Discreto
•1 grau de liberdade
•Aquisitor de caracteres/ Aquisitor de pontos/ Aquisitor de valores/ Localizador (posição relativa)/Selecionador/ Identificador

Dispositivos de Entrada
Tablet ou Pad digitalizador

•Controle por indução eletromagnética
•Contínuo
•2 graus de liberdade
•Aquisitor de pontos/Localizador (posição absoluta)/Selecionador/ Identificador/Aquisitor de valores (ângulo)/ Aquisitor de caracteres (*handwriting*)

Dispositivos de Entrada
Joystick

•Controle por movimento
•Contínuo
•2 graus de liberdade
•Localizador (posição relativa)/ Selecionador/ Identificador



Dispositivos de Entrada
Spaceball

•Controle por movimento
•Contínuo
•6 graus de liberdade
•Aquisitor de pontos/Localizador (posição relativa)/Selecionador/ Identificador

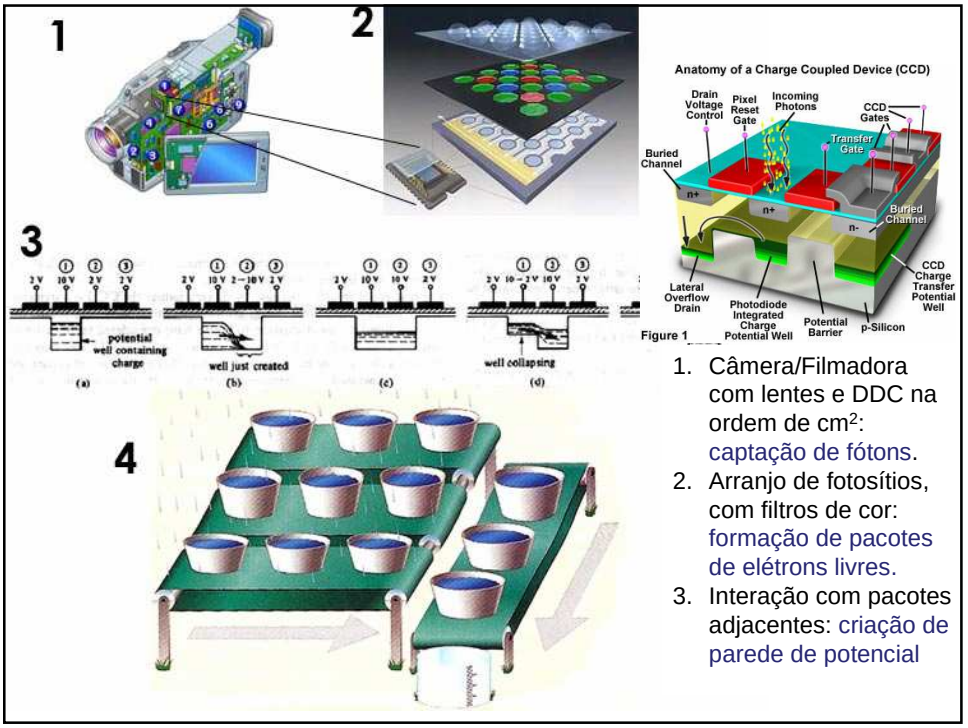
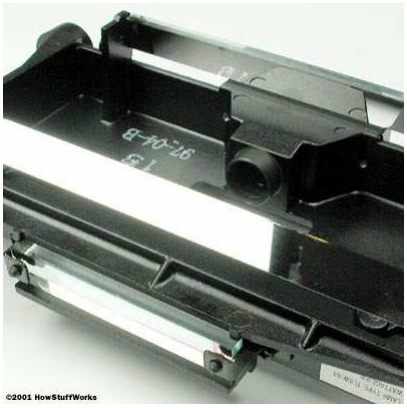


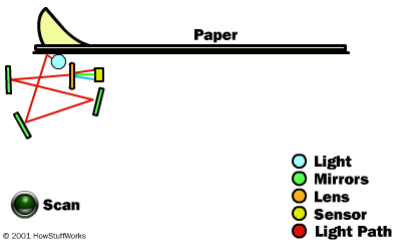
Figure 1

1. Câmera/Filmadora com lentes e DDC na ordem de cm^2: captação de fótons.
2. Arranjo de fotosítios, com filtros de cor: formação de pacotes de elétrons livres.
3. Interação com pacotes adjacentes: criação de parede de potencial

Dispositivos de Entrada

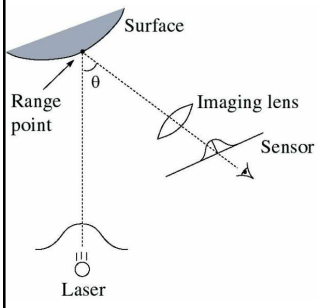
Scanner (OCR)

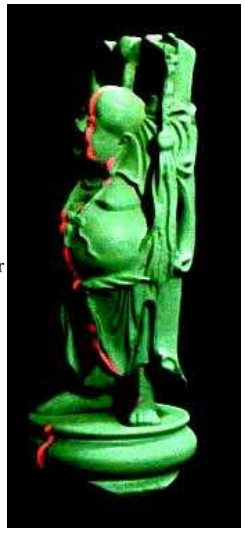


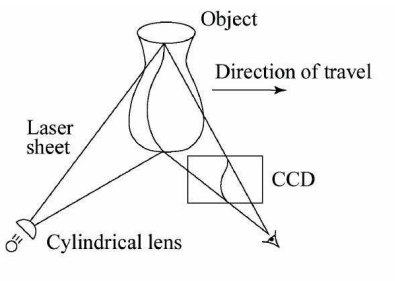


Reconhecimento óptico de caracteres

Dispositivos de Entrada
Sensor de Profundidade

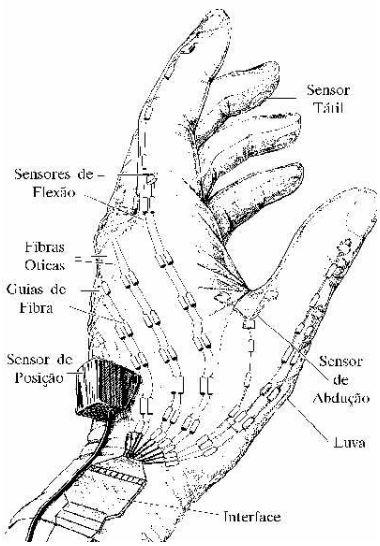






Dispositivos Físicos
Data Glove

- Controle por movimento
- Contínuo
- graus de liberdade dependentes dos sensores
- Aquisitor de pontos/Localizador (posição relativa)/Selecionador/ Identificador



Dispositivos de Entrada Lógicos
Função Lógica

- Aquisitor de cadeias de caracteres (*string*)
  - Cadeia de caracteres
- Localizador (*locator*)
  - Posição de um ponto no espaço
- Identificador (*pick*)
  - Um elemento gráfico dentre os exibidos
- Selecionador (*choice*)
  - Uma dentre as alternativas pré-estabelecidas
- Aquisitor de valores escalares (*valuator*)
  - Um valor
- Aquisitor de seqüências de segmentos (*stroke*)
  - Seqüência de pontos

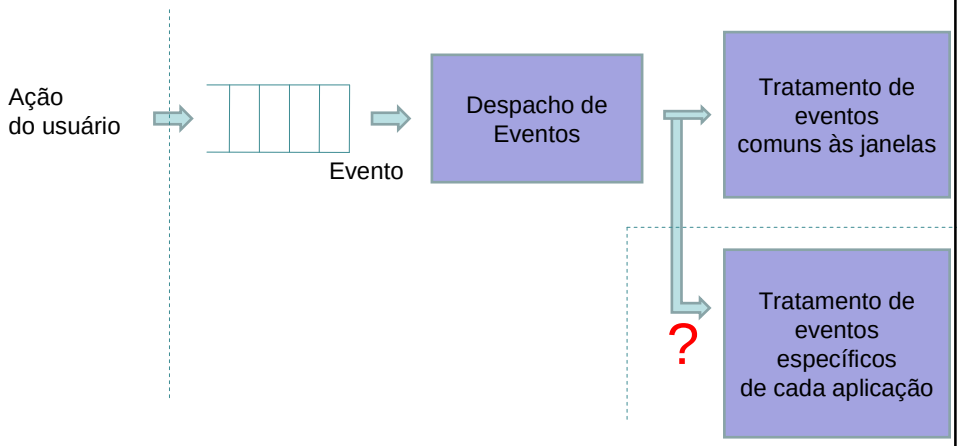
Processamento de Eventos de Entrada
MainEventLoop

A qual janela deve ser despachado o evento?

Como o tratamento pode ser personalizado para cada aplicativo?

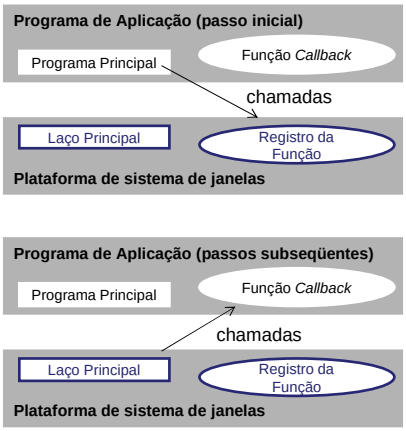
Sistema de janelas

Estratégia de Tratamento de Eventos



Programação Orientada a Eventos

Modelo *Callback*



Quantas funções devem ser previstas?

Aplicativos Gráficos Interativos

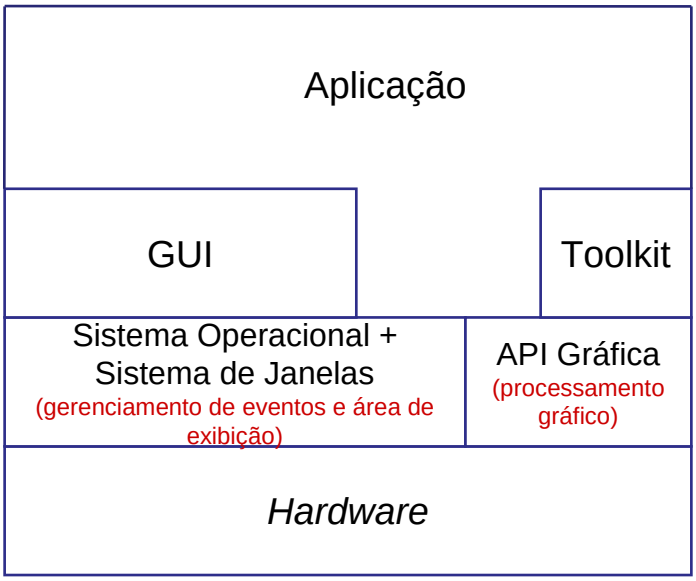
O **Teorema de Pappus** (atribuido a Pappus de Alexandria) afirma que dado um conjunto de pontos ***colineares*** *a0*, *a1*, *a2*, e um outro conjunto de pontos lineares *b0*, *b1*, *b2*, então os pontos de intesecção *P1*, *P2*, *P0* dos pares de linha *a1b0* e *a0b1*, *a0b2* e *a2b0*, *a1b2* e *a2b1* são colineares.

http://i33www.ira.uka.de/applets/mocca/html/noplugin/Pappus/AppPappus/index.html

Algumas Diretrizes para Desenvolvimento de Aplicativos Gráficos Interativos

- Representação visual compatível com o contexto
- Consistência em *look* e *feel*
- Realimentação às ações dos usuários, respeitando os limites aceitáveis dos tempos de resposta
  - Até 0.1s: nenhuma realimentação especial é necessária
  - Até 1s: o usuário ainda não perde a noção de fluidez
  - Até 10s: o usuário ainda aceita a espera
  - Mais de 10s: realimentação especial é necessária
- Minimização da probabilidade de ocorrência de erros, habilitando somente ações válidas
- Minimização do esforço de memorização
- Inclusão de ações: *Undo, Redo, Reset, Cancel*, ou similares.

Aplicativo Gráfico Interativo



APIs para Desenvolvimento de Aplicativos Gráficos Interativos

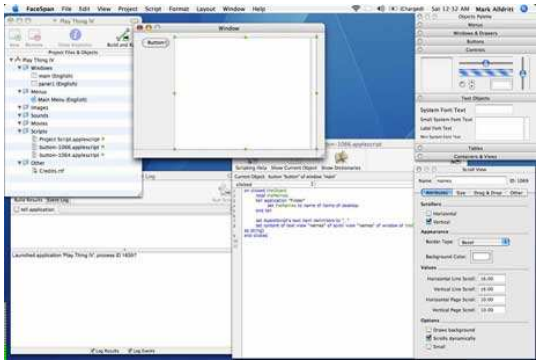
- API Gráfica (3D)
  - OpenGL
  - Direct3D

- GUI provida de área de desenho OpenGL
  - GLUT
  - Fast Light Toolkit (FLTK)
  - wxWidgets
  - Qt – Trolltech
  - GTK
  - GLUI

Interface Gráfica de Usuário

GUI

Componentes de interface (*Widgets*) providos de geometria (*look*) e comportamento (*feel*) próprios.

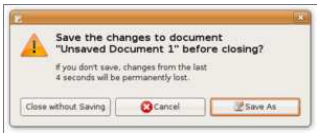


Alguns Exemplos de Componentes

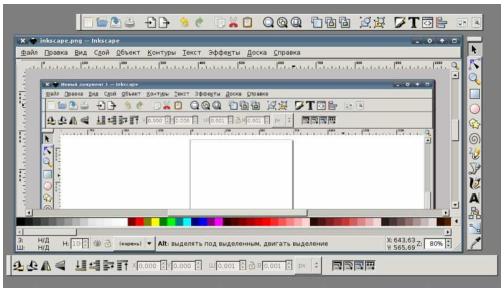
Button

List box

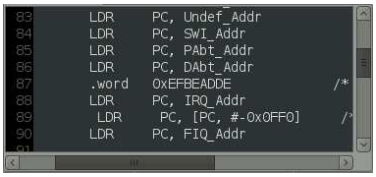
Spinner



Dialog box

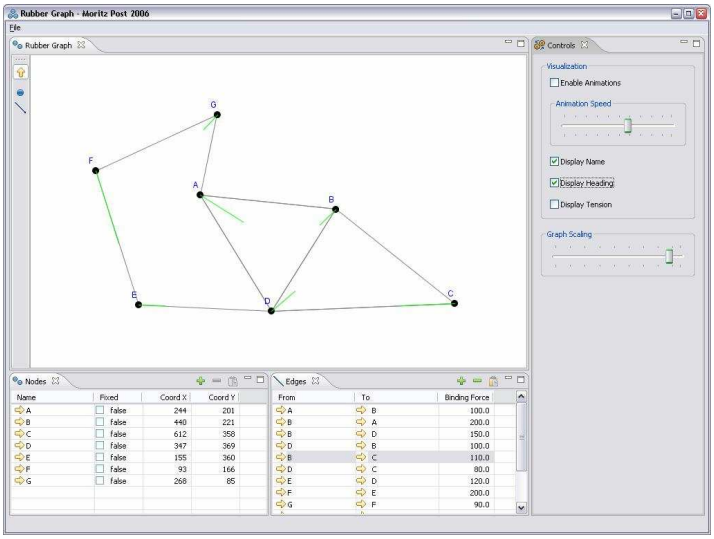


Toolbar + menubar

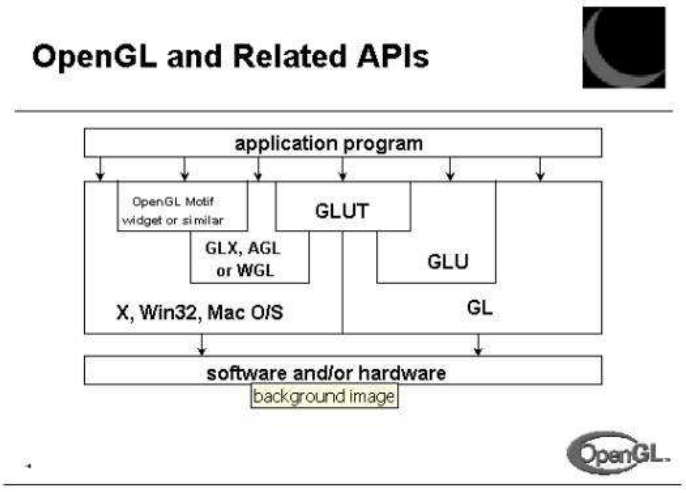


Scrollbar

Área de Desenho/*Canvas*

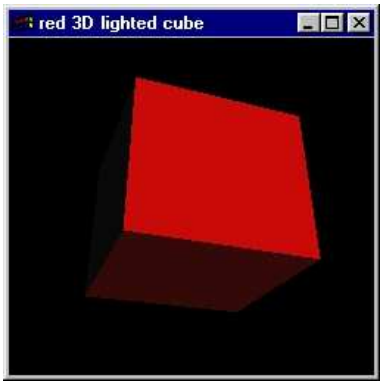


APIs de OpenGL



http://www.cosc.brocku.ca/Offerings/3P98/course/lectures/OpenGL/glut_ref.html

Um Aplicativo




```
int main(int argc, char **argv) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE |
            GLUT_RGB | GLUT_DEPTH);
    glutCreateWindow("red 3D lighted cube");
    glutDisplayFunc(display);
    init();
    glutMainLoop();
    return 0;
    /* ANSI C requires main to return int. */
}
```

http://www.opengl.org/resources/code/samples/glut_examples/examples/examples.html

```
void init(void) {
  /* Setup cube vertex data. */
  v[0][0] = v[1][0] = v[2][0] = v[3][0] = -1;
  v[4][0] = v[5][0] = v[6][0] = v[7][0] = 1;
  v[0][1] = v[1][1] = v[4][1] = v[5][1] = -1;
  v[2][1] = v[3][1] = v[6][1] = v[7][1] = 1;
  v[0][2] = v[3][2] = v[4][2] = v[7][2] = 1;
  v[1][2] = v[2][2] = v[5][2] = v[6][2] = -1;
  /* Enable a single OpenGL light. */
  glLightfv(GL_LIGHT0, GL_DIFFUSE,
            light_diffuse);
  glLightfv(GL_LIGHT0, GL_POSITION, l
            ight_position);
  glEnable(GL_LIGHT0);
glEnable(GL_LIGHTING);
  // Use depth buffering for hidden surface
  //elimination.
  glEnable(GL_DEPTH_TEST);
  setView();
}
```

```
setView(void) {
  /* Setup the view of the cube. */
  glMatrixMode(GL_PROJECTION);
  gluPerspective(
   /* field of view in degree*/ 40.0,
   /* aspect ratio */ 1.0,
   /* Z near */ 1.0,
   /* Z far */ 10.0);
   glMatrixMode(GL_MODELVIEW);
   gluLookAt(
     0.0, 0.0, 5.0,  /* eye is at (0,0,5) */
     0.0, 0.0, 0.0, /* center is at (0,0,0) */
     0.0, 1.0, 0.); /* up is in +Y direction */
   //Adjust cube position to be asthetic
   //angle.
   glTranslatef(0.0, 0.0, -1.0);
   glRotatef(60, 1.0, 0.0, 0.0);
   glRotatef(-20, 0.0, 0.0, 1.0);
}
```

```
void display(void) {
glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
  drawBox();
  glutSwapBuffers();
}
 void drawBox(void) {
       int i; for (i = 0; i < 6; i++) {
              glBegin(GL_QUADS);
              glNormal3fv(&n[i][0]);
              glVertex3fv(&v[faces[i][0]][0]);
              glVertex3fv(&v[faces[i][1]][0]);
              glVertex3fv(&v[faces[i][2]][0]);
              glVertex3fv(&v[faces[i][3]][0]); glEnd();
       }
 }
```