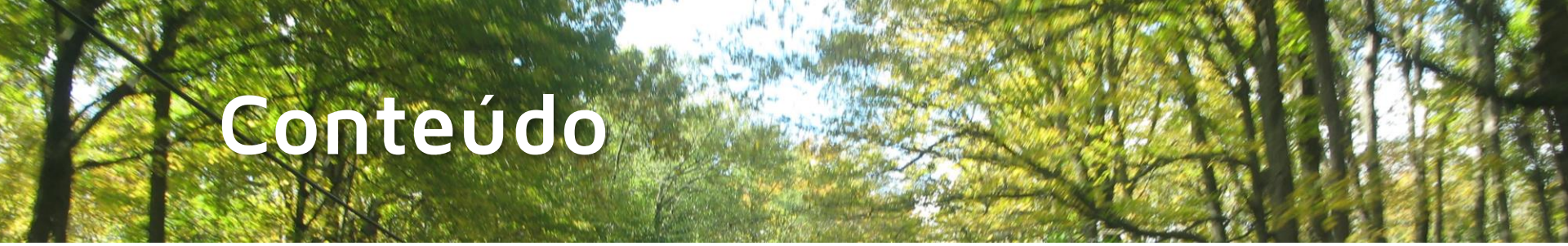
Módulo 1

Sistemas Gráficos e Interação
Instituto Superior de Engenharia do Porto

Filipe Pacheco
ffp@isep.ipp.pt

Introdução ao OpenGL



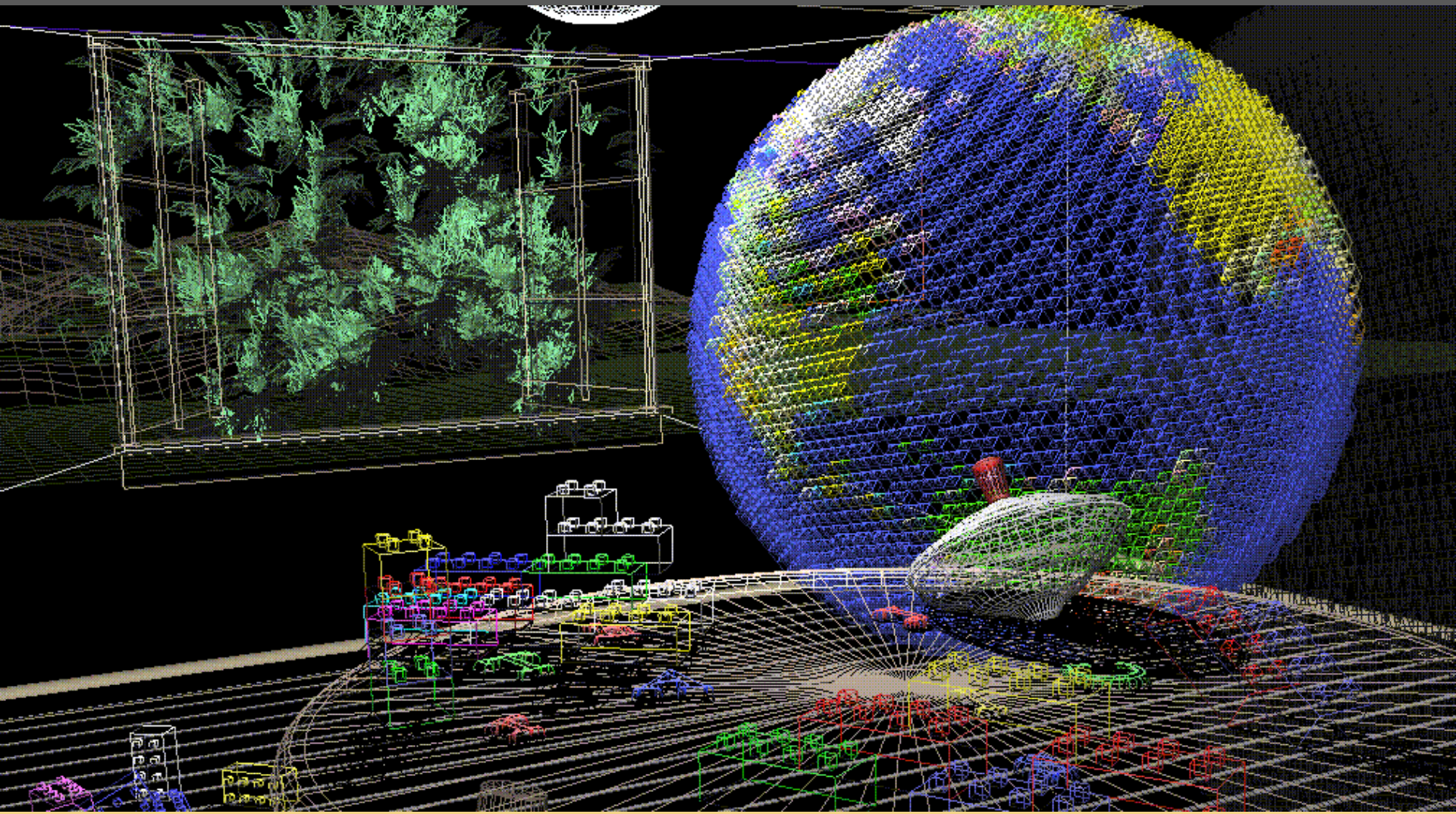
Conteúdo

- Introdução
- Tipos de dados
- Nomenclatura de funções
- Esqueleto de programa
  - Atividades básicas
- Esqueleto de programa usando GLUT
  - Inicialização
  - *Callback* de desenho
- Exemplos de aplicações

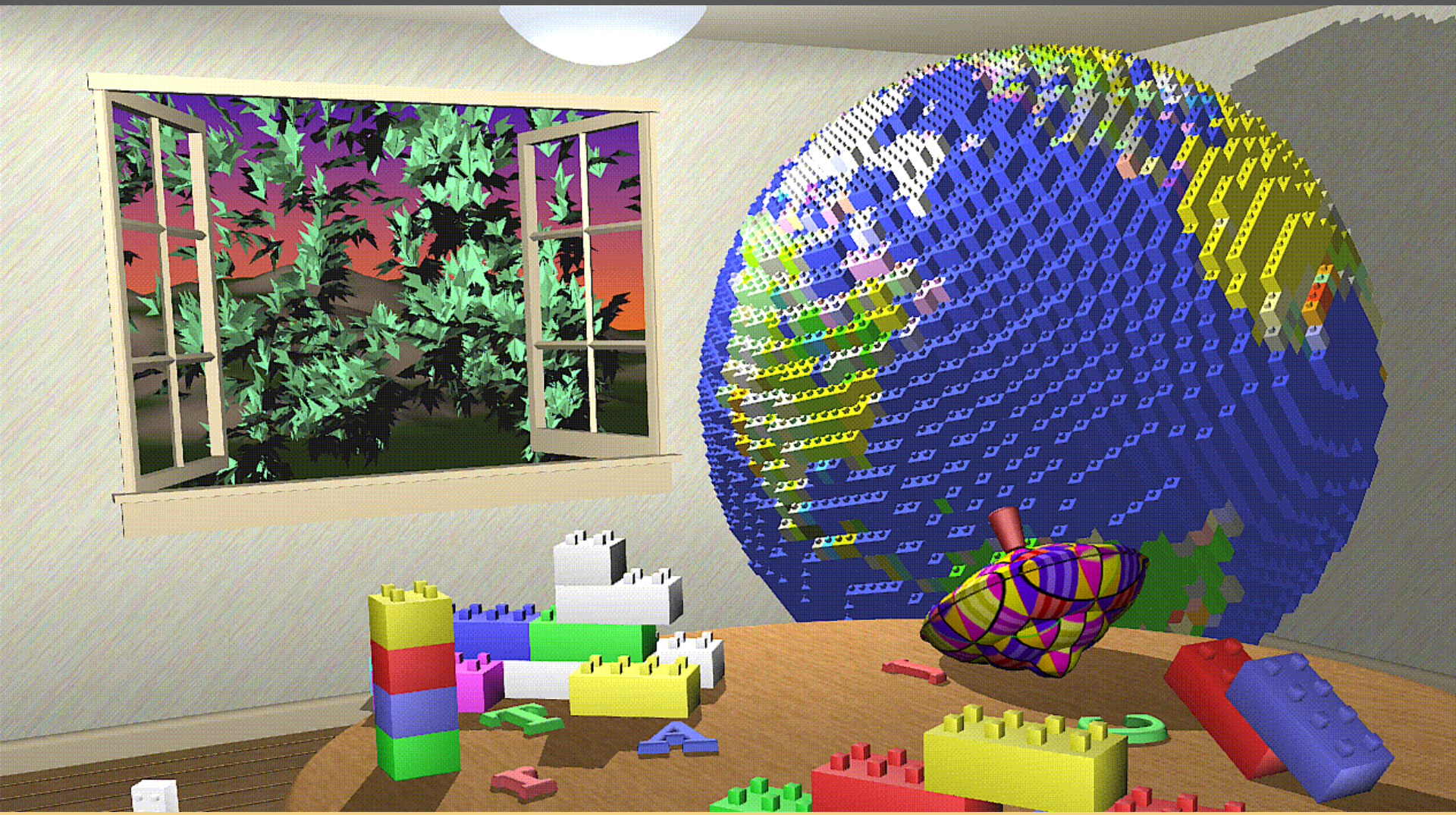
OpenGL

- **Open G**raphics **L**ibrary
- API genérica de gráficos 2D/3D
  - Apenas contém primitivas simples
  - Modelos complexos construídos com base nestas primitivas
- Independente do sistema operativo
- Independente do sistema gestor de janelas

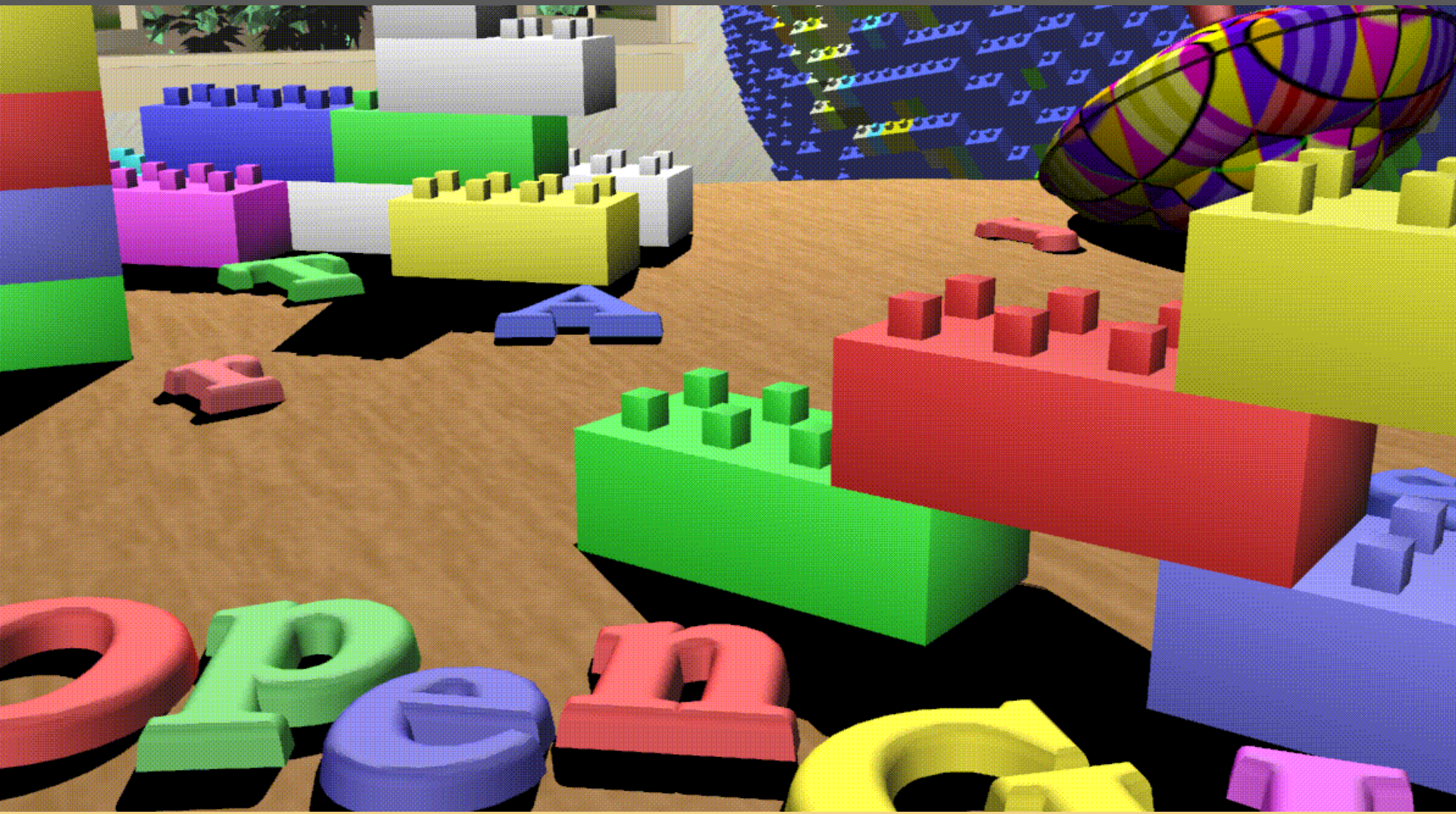
Uma cena: wireframe



Sombras e iluminação



Pormenor



Efeitos ambiente

"Bits and bytes"

- OpenGL
- GLU – OpenGL Utility library
- GLUT – OpenGL Utility Toolkit
- GLX / WGL / CGL – OpenGL for X-Windows/OpenGL for Windows/OpenGL for Mac OS X
- Java bindings
- GlAx – OpenGL ActiveX
- CsGL – OpenGL for C#
- WebGL – OpenGL for HTML 5

Tipos de dados

Suffix	Data Type	Typical Corresponding C-Language Type	OpenGL Type Definition
b	8-bit integer	signed char	GLbyte
s	16-bit integer	short	GLshort
i	32-bit integer	int or long	GLint, GLsizei
f	32-bit floating-point	float	GLfloat, GLclampf
d	64-bit floating-point	double	GLdouble, GLclampd
ub	8-bit unsigned integer	unsigned char	GLubyte, GLboolean
us	16-bit unsigned integer	unsigned short	GLushort
ui	32-bit unsigned integer	unsigned int or unsigned long	GLuint, GLenum, GLbitfield

Nomenclatura de funções

- Prefixos no nome das funções
  - gl ➔ OpenGL
  - glu ➔ *OpenGL* Utility Library
  - glut ➔ *OpenGL* Utility Toolkit
  - glaux ➔ *OpenGL* Auxiliary Library (deprecated)
- { gl | glu | glut } FUNC [{ 1 | 2 | 3 | 4 } TYPE [v]]
  - TYPE := b | d | f | i | s | ub | ui | us
- Exemplos:
  - **gl**Color**3b**, **gl**Color**3f**, **gl**Color**4d**, **gl**Color**4dv**
  - **gl**LoadMatrix**d**, **gl**LoadMatrix**f**
  - **glu**LookAt
  - **glut**SwapBuffers

Esqueleto de programa

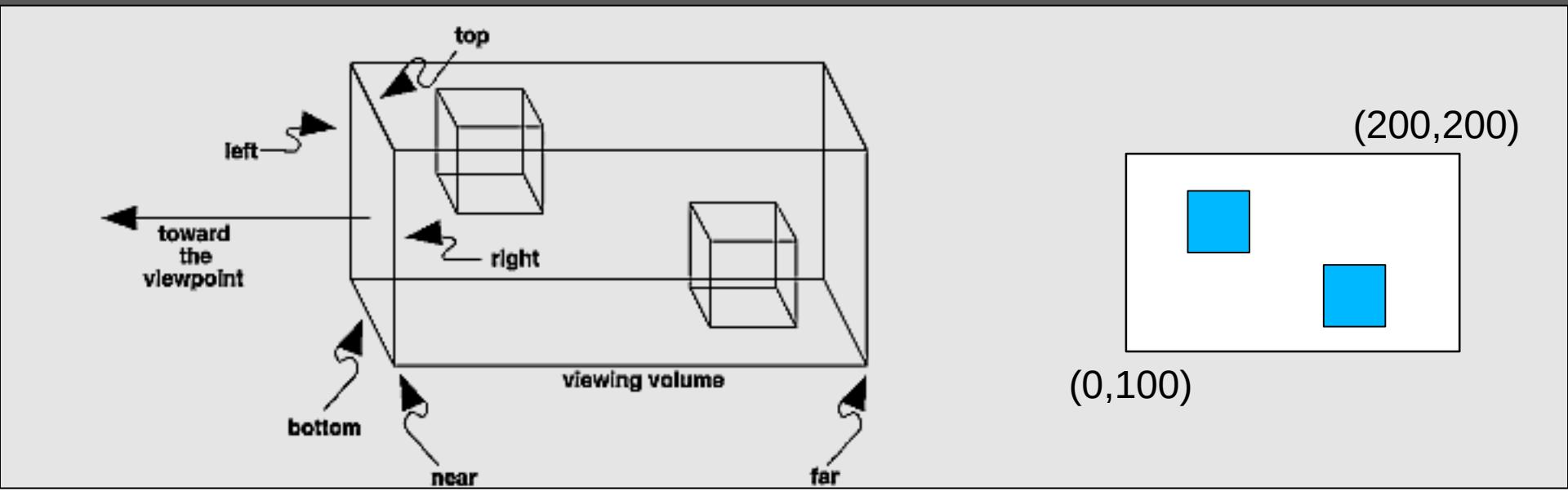
```
#include <whateverYouNeed.h>
int main()
{
    InitializeAWindowPlease();
    glClearColor(0.0, 0.0, 0.0, 0.0);
    glClear(GL_COLOR_BUFFER_BIT);
    glMatrixMode(GL_PROJECTION)
    glLoadIdentity();
    glOrtho(0.0, 1.0, 0.0, 1.0, -1.0, 1.0);
    glColor3f(1.0, 1.0, 1.0);
    glBegin(GL_POLYGON);
        glVertex3f (0.25, 0.25, 0.0);
        glVertex3f (0.75, 0.25, 0.0);
        glVertex3f (0.75, 0.75, 0.0);
        glVertex3f (0.25, 0.75, 0.0);
    glEnd();
    glFlush();
    UpdateTheWindowAndCheckForEvents();
    return 0;
}
```

Limpar ecrã

Definir sistema de coordenadas

Desenhar

Sistema de coordenadas



- gluOrtho2D(0.0, 200.0, 100.0, 200.0)

left

right

bottom

top

Por omissão near=1.0 far=-1.0

Cor

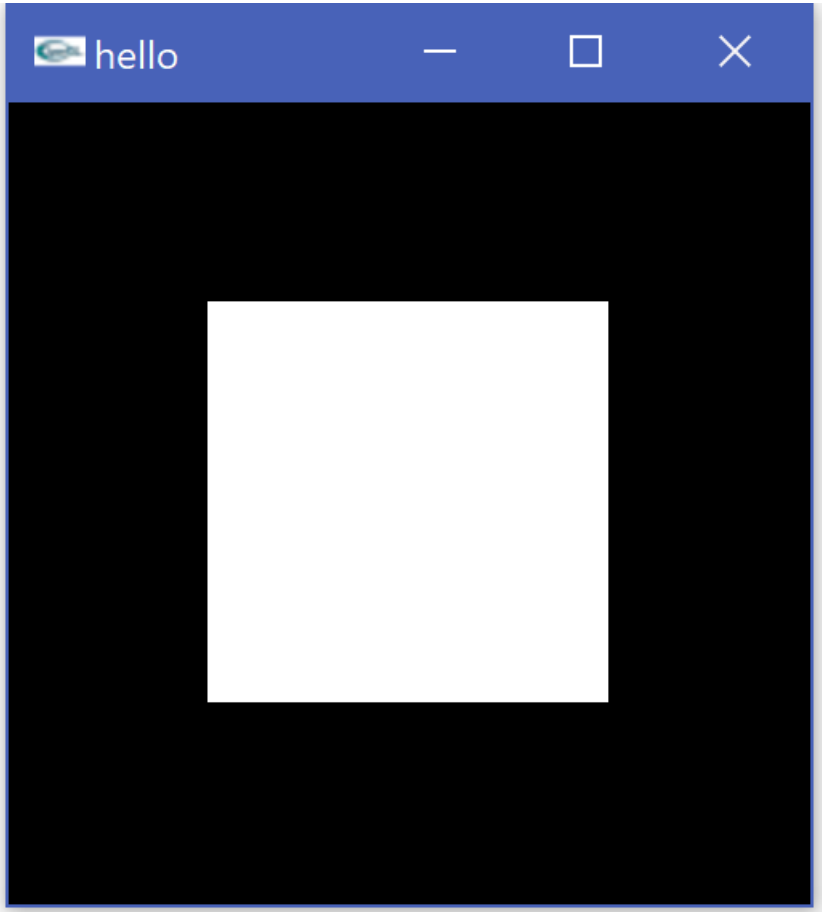
- **RGBA**
  glColor{3|4}
  red + green + blue [+ alfa]
  alfa : usado para *blending* e transparências (falaremos futuramente)
  [0.0 , 1.0]
- **Index** (*color map*)
  Utiliza *color map* do sistema gestor de janelas
  Um índice para o mapa em vez dos componentes de cor
- **Exemplos**

```
glColor3f(0.0, 0.0, 0.0);    /* black     */
glColor3f(1.0, 0.0, 0.0);    /* red       */
glColor3f(0.0, 1.0, 0.0);    /* green     */
glColor3f(1.0, 1.0, 0.0);    /* yellow    */
glColor3f(0.0, 0.0, 1.0);    /* blue      */
glColor3f(1.0, 0.0, 1.0);    /* magenta   */
glColor3f(0.0, 1.0, 1.0);    /* cyan      */
glColor3f(1.0, 1.0, 1.0);    /* white     */
```

Desenhar

- glBegin(mode) / glEnd()
  - Define um objeto da cena
  - Por exemplo: polígono (GL_POLYGON)
- glVertex...()
  - Define um vértice do objeto a desenhar
- glFlush()
  - Força o pipeline do OpenGL a terminar o processamento e desenhar os pixels no ecrã

Demo



Exemplos