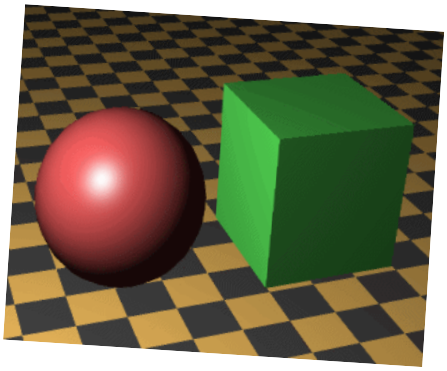
MANCHESTER 1824

COMP27112

Computer Graphics and Image Processing



2: Introducing image synthesis

Toby.Howard@manchester.ac.uk

1

MANCHESTER 1824

Introduction

In these notes we'll cover:

- Some orientation – how did we get here?
- Graphics system architecture
- Overview of OpenGL / GLU / GLUT

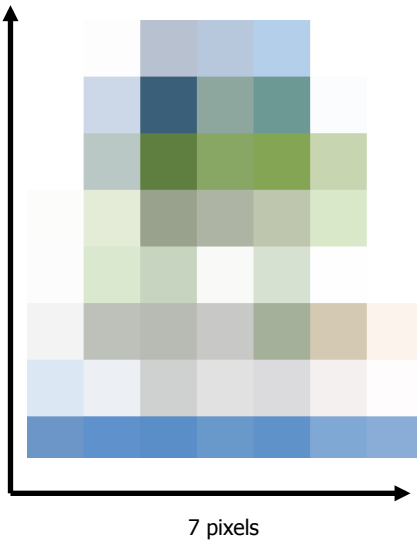
- These notes are intended to be read in conjunction with the OpenGL programming manual – that's where all the detailed programming information is

2

MANCHESTER 1824

Raster graphics displays

- Raster graphics uses a 2D array of pixels (screens) or dots (printers)
- So images must be sampled
- The more samples, the better the fidelity
- But always an approximation



3

MANCHESTER 1824

Raster graphics displays

- Raster graphics uses a 2D array of pixels (screens) or dots (printers)
- So images must be sampled
- The more samples, the better the fidelity
- But always an approximation

4

MANCHESTER 1824

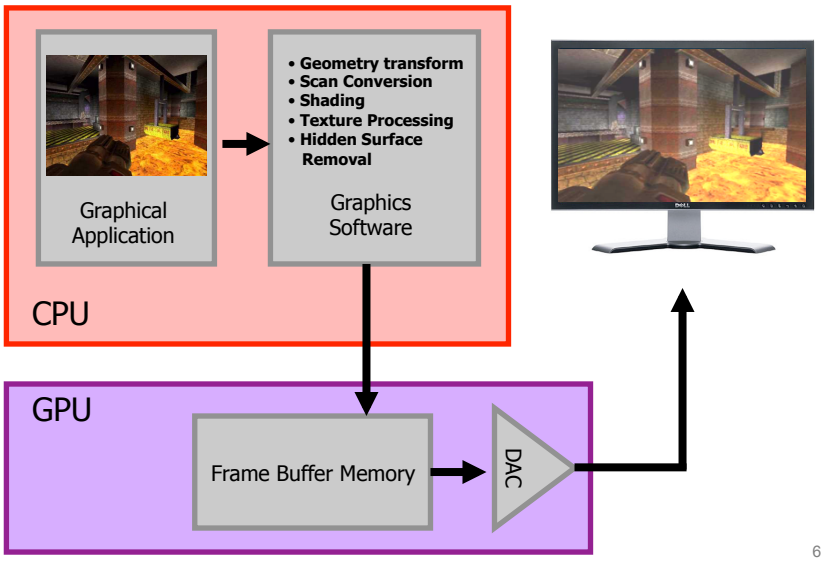
Alternative: Vector Displays

- The first type of graphical display
- Cathode Ray Tube and controller
- Draws vectors… essentially lines from point A to point B
- Usually monochrome (apparent colour here is done with transparent plastic overlays!)

5

MANCHESTER 1824

Basic graphics system architecture

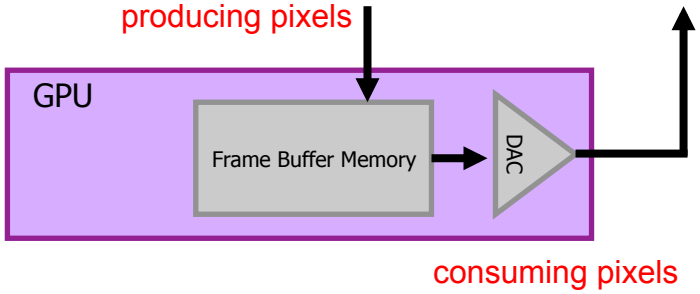


6

MANCHESTER 1824

Basic graphics system architecture

This is the classic producer/consumer situation

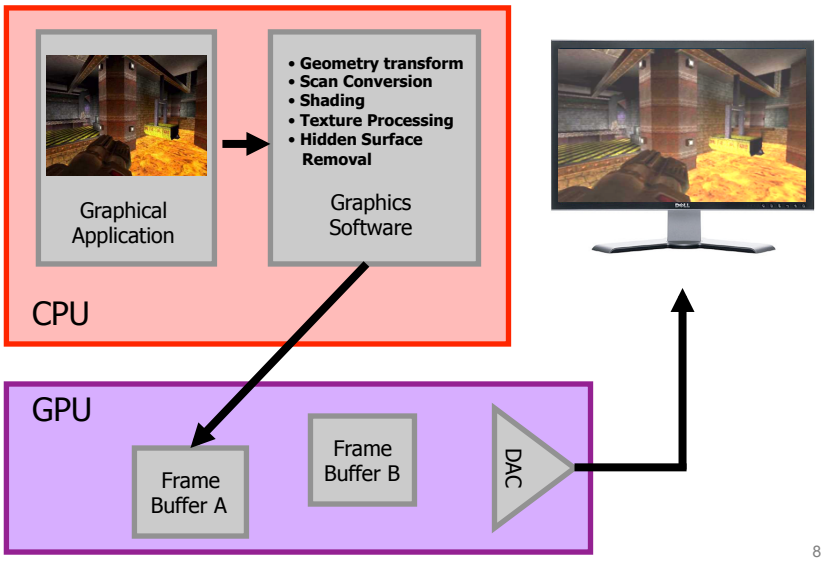


How do we **synchronise** the two activities?

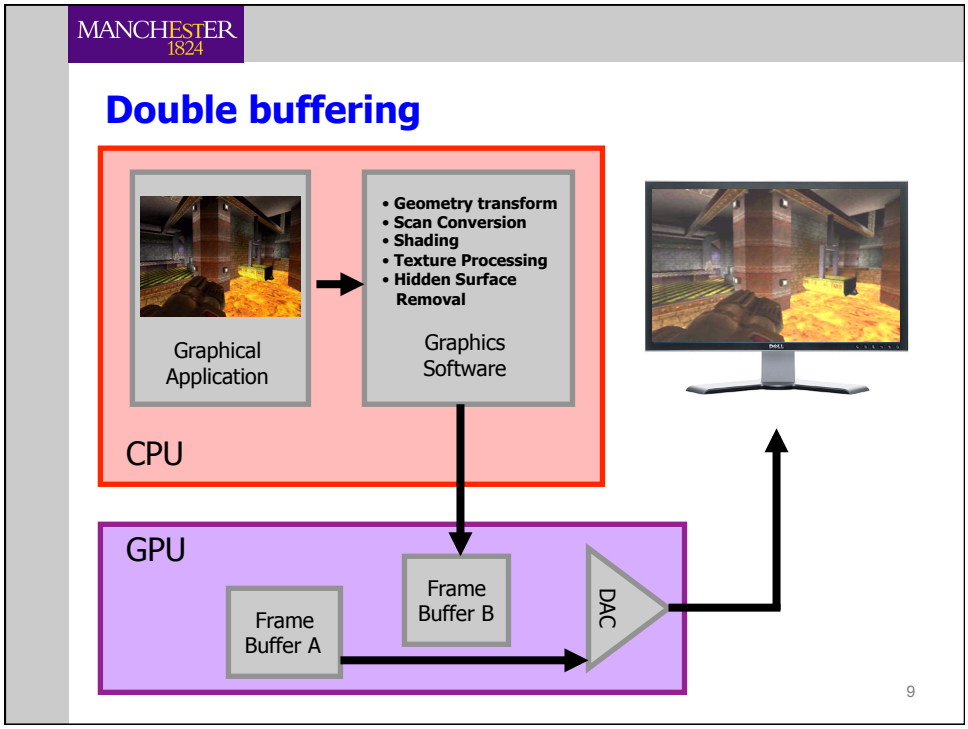
7

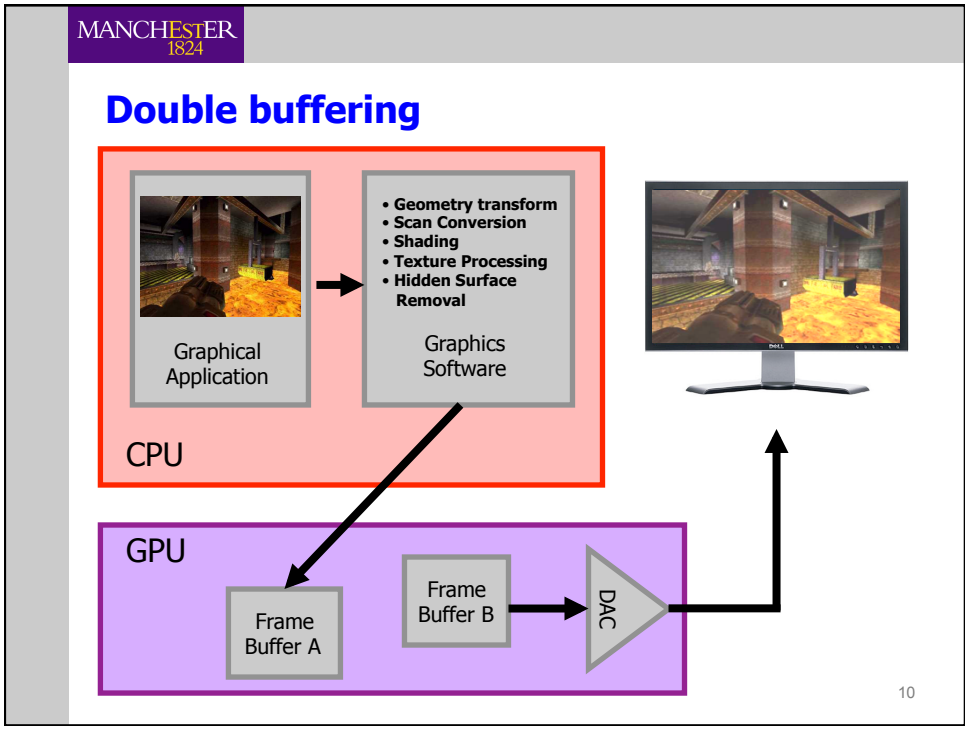
MANCHESTER 1824

Double buffering



8


MANCHESTER 1824
Double buffering
• Geometry transform
• Scan Conversion
• Shading
• Texture Processing
• Hidden Surface Removal
Graphical Application
Graphics Software
CPU
GPU
Frame Buffer A
Frame Buffer B
DAC
9


MANCHESTER 1824
Double buffering
• Geometry transform
• Scan Conversion
• Shading
• Texture Processing
• Hidden Surface Removal
Graphical Application
Graphics Software
CPU
GPU
Frame Buffer A
Frame Buffer B
DAC
10

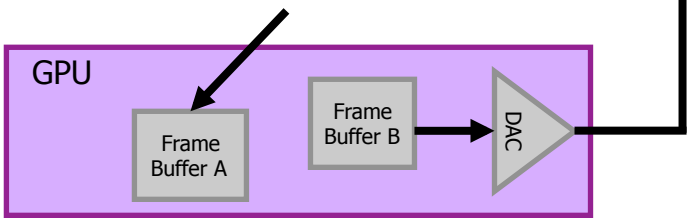
MANCHESTER 1824

Double buffering

The swap between the buffers takes place during the vertical retrace time of the monitor (aka VBLANK) – the time between the end of drawing one frame, and beginning to draw the next.

This time is a function of the refresh rate of the monitor (typically 75Hz).

In OpenGL, the `glutSwapBuffers()` command requests that a swap be scheduled for the next retrace.

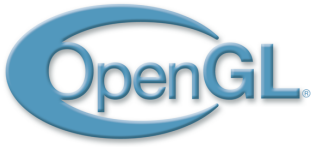


11

MANCHESTER 1824

OpenGL

- Open Graphics Library

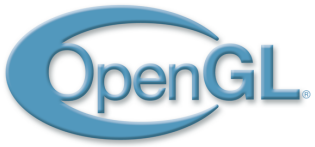


- platform/language-independent
- origins in Silicon Graphics' proprietary GL
- OpenGL first released 1992
- Today: open standard for portable graphics programming (www.opengl.org)

12

MANCHESTER 1824

OpenGL vs. DirectX



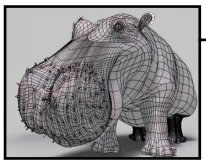
- OpenGL
  - Open standard, run by the **Khronos Group**
  - Cross-architecture (Win/Mac/Linux…)
- DirectX
  - Owned by Microsoft

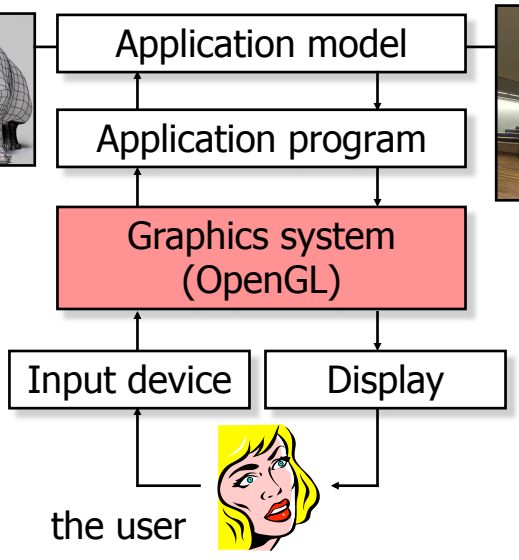
- Functionally, the two systems are broadly similar
- For an in-depth comparison, see http://en.wikipedia.org/wiki/Comparison_of_OpenGL_and_Direct3D

13

MANCHESTER 1824

Where OpenGL fits in





14

MANCHESTER 1824

The OpenGL API

- OpenGL is a **specification** of an Application Programmer's Interface (API)
- A set of functions for doing 3D computer graphics
- used in industry, engineering, CAD, CGI/ SFX, research, games, education – everywhere

15

MANCHESTER 1824

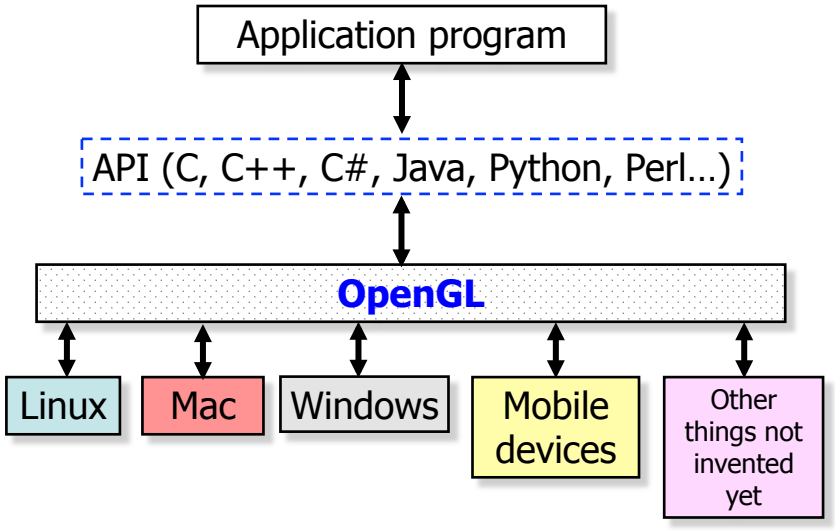
OpenGL evolution

- OpenGL functionality has evolved over time:
  - OpenGL v1, v2: "fixed pipeline" – fixed functionality
  - OpenGL v3+: "programmable pipeline" – extensible functionality: programmers write "shader" micro-programs
  - OpenGL 4.5 (Aug 2014)
- In this introductory course we use the fixed pipeline
  - **pro**: programming is simpler, more built-in support
  - **con**: lack of flexibility, cannot fully exploit GPU

16

MANCHESTER 1824

Portability

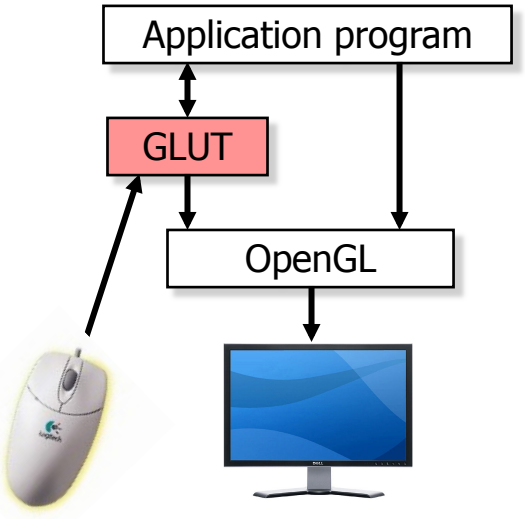


17

MANCHESTER 1824

OpenGL and interaction

- OpenGL only generates pixels
- it **doesn't** handle interaction devices
- for interaction, we use an additional library (called GLUT, see later)



18

MANCHESTER 1824

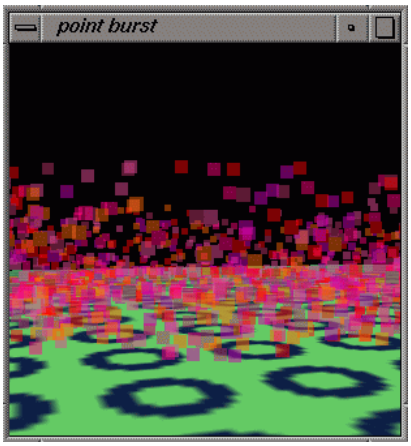
OpenGL graphics

- The main features of OpenGL
  - 3D graphics (points, lines, polygons…)
  - coordinate transformations
  - a camera for viewing
  - hidden surface removal
  - lighting and shading
  - texturing
  - pixel (image) operations
- We'll take a whirlwind tour

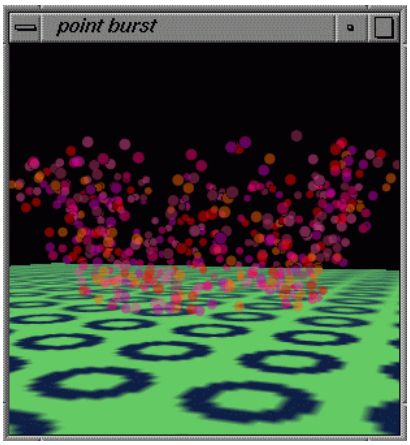
19

MANCHESTER 1824

Points



Regular points



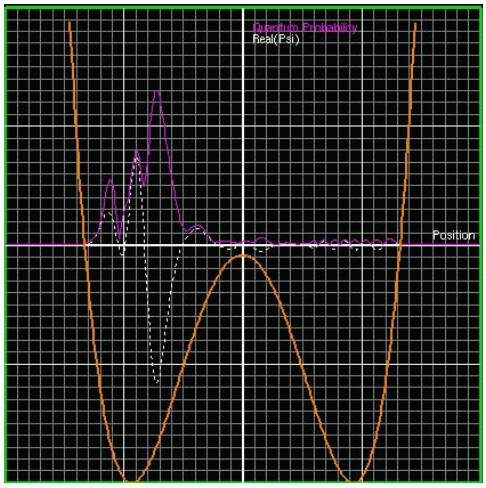
Anti-aliased points

20

MANCHESTER 1824

Lines

- Lines, like all graphical primitives, have two kinds of property:
  - **geometry** (shape)
  - **attributes** (appearance)

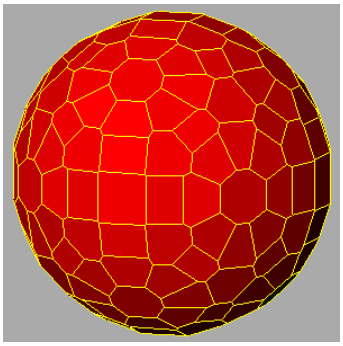


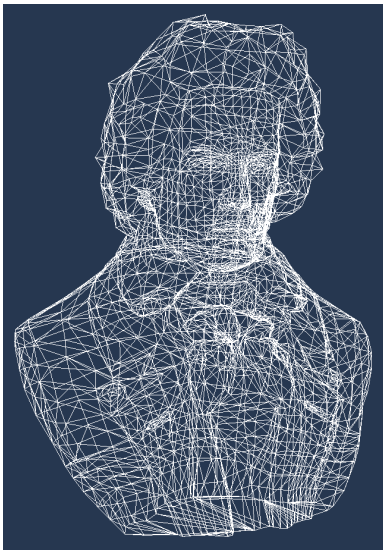
21

MANCHESTER 1824

Polygons

- Triangles
- Quadrilaterals
- Convex polygons



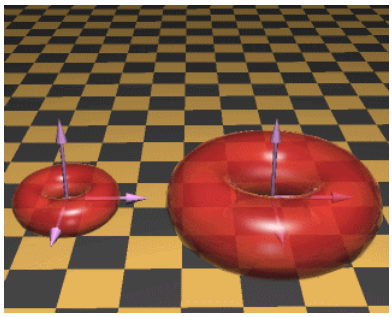


22

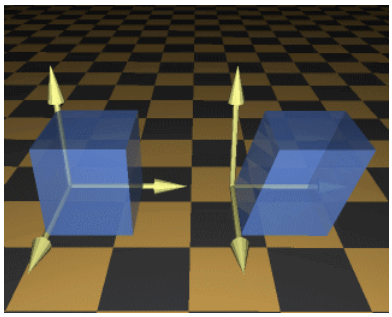
MANCHESTER 1824

Transformations

- OpenGL expresses coordinate transformations as 4x4 homogeneous matrices



scaling

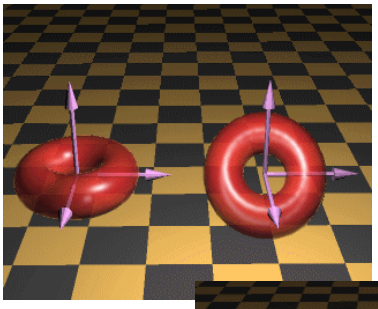


shearing

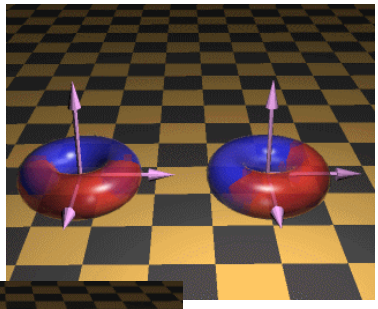
23

MANCHESTER 1824

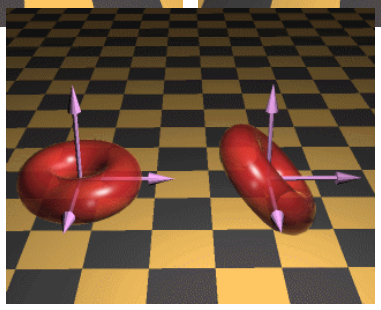
Transformations



rotation about x



rotation about y



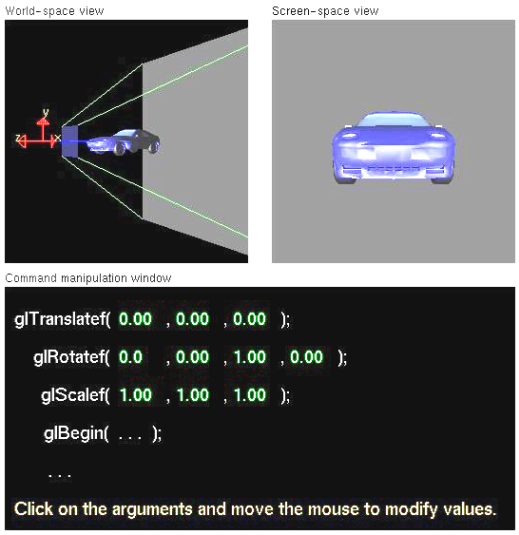
rotation about z

24

MANCHESTER 1824

Transformations

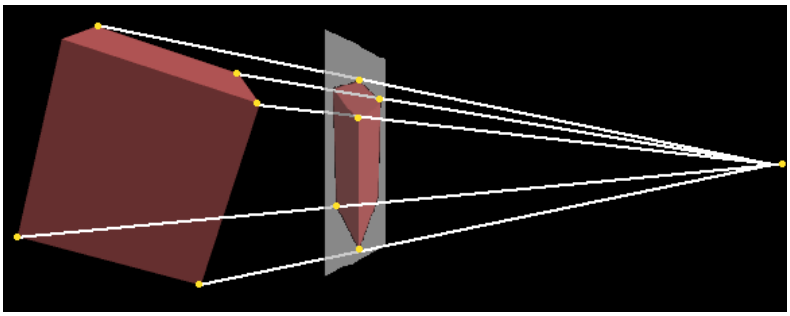
- Transformations can be combined and nested
  - Allowing complex objects to be assembled from simpler parts



MANCHESTER 1824

The camera model

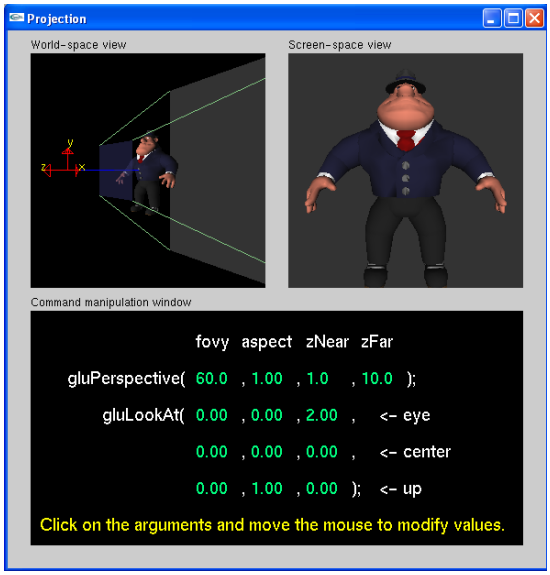
- Creating 2D views of 3D scenes
  - perspective projection
  - parallel projection



26

MANCHESTER 1824

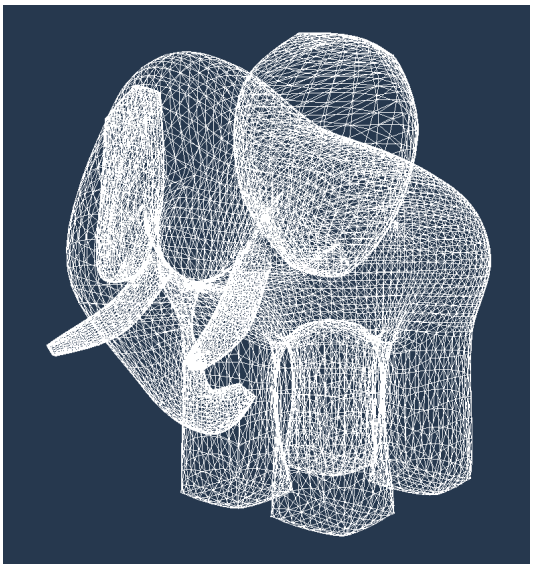
The camera model



27

MANCHESTER 1824

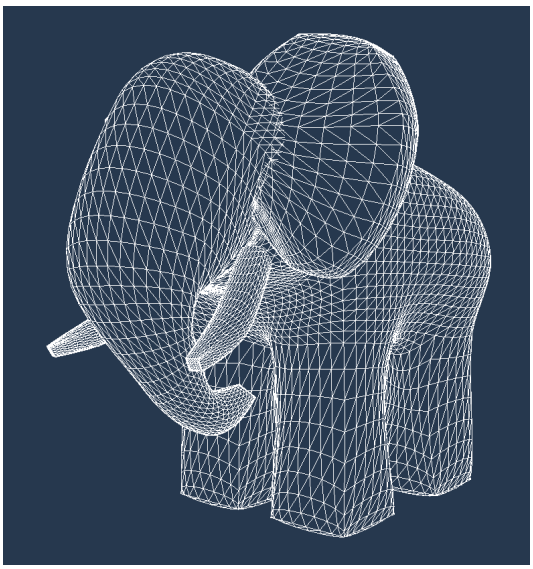
Hidden-surface removal



28

MANCHESTER 1824

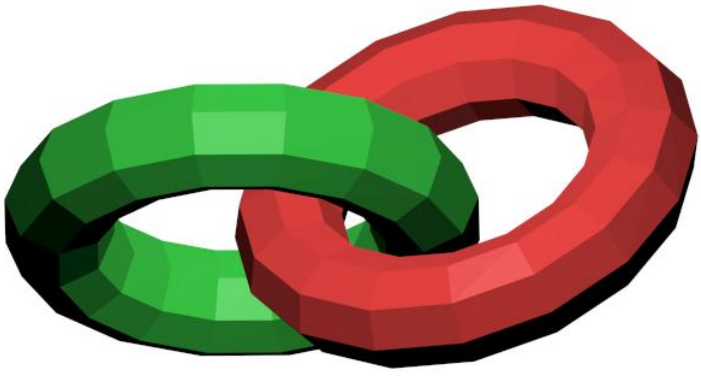
Hidden-surface removal



29

MANCHESTER 1824

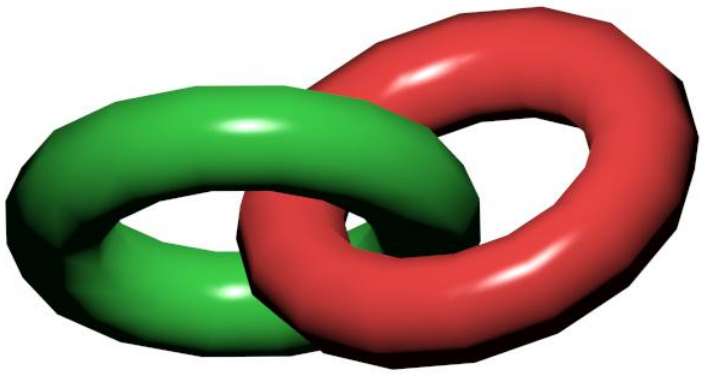
Lighting and shading



30

MANCHESTER 1824

Lighting and shading

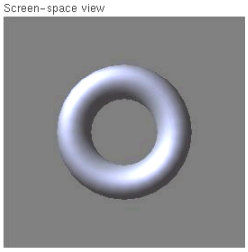


31

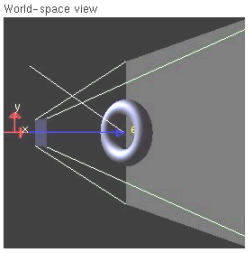
MANCHESTER 1824

Lighting and shading

Screen-space view



World-space view



Command manipulation window

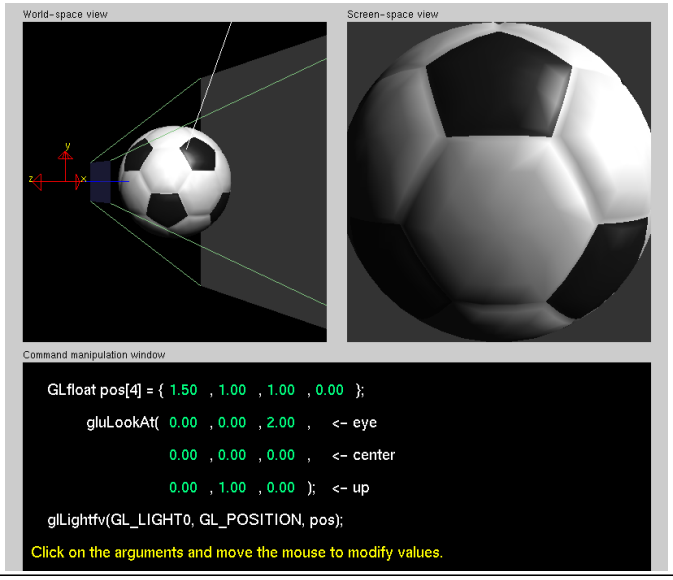
```
GLfloat light_pos[ ] = {  -2.00 , 2.00  , 2.00  , 1.00  };
GLfloat light_Ka[ ] = {  0.00  , 0.00  , 0.00  , 1.00  };
GLfloat light_Kd[ ] = {  1.00  , 1.00  , 1.00  , 1.00  };
GLfloat light_Ks[ ] = {  1.00  , 1.00  , 1.00  , 1.00  };

glLightfv(GL_LIGHT0, GL_POSITION, light_pos);
glLightfv(GL_LIGHT0, GL_AMBIENT, light_Ka);
glLightfv(GL_LIGHT0, GL_DIFFUSE, light_Kd);
glLightfv(GL_LIGHT0, GL_SPECULAR, light_Ks);

GLfloat material_Ka[ ] = { 0.11  , 0.06  , 0.11  , 1.00  };
GLfloat material_Kd[ ] = { 0.43  , 0.47  , 0.54  , 1.00  };
GLfloat material_Ks[ ] = { 0.33  , 0.33  , 0.52  , 1.00  };
GLfloat material_Ke[ ] = { 0.00  , 0.00  , 0.00  , 0.00  };
GLfloat material_Se =  10  ;

glMaterialfv(GL_FRONT, GL_AMBIENT, material_Ka);
glMaterialfv(GL_FRONT, GL_DIFFUSE, material_Kd);
glMaterialfv(GL_FRONT, GL_SPECULAR, material_Ks);
glMaterialfv(GL_FRONT, GL_EMISSION, material_Ke);
glMaterialfv(GL_FRONT, GL_SHININESS, material_Se);
```

Click on the arguments and move the mouse to modify values.

32

MANCHESTER 1824

Light position



33

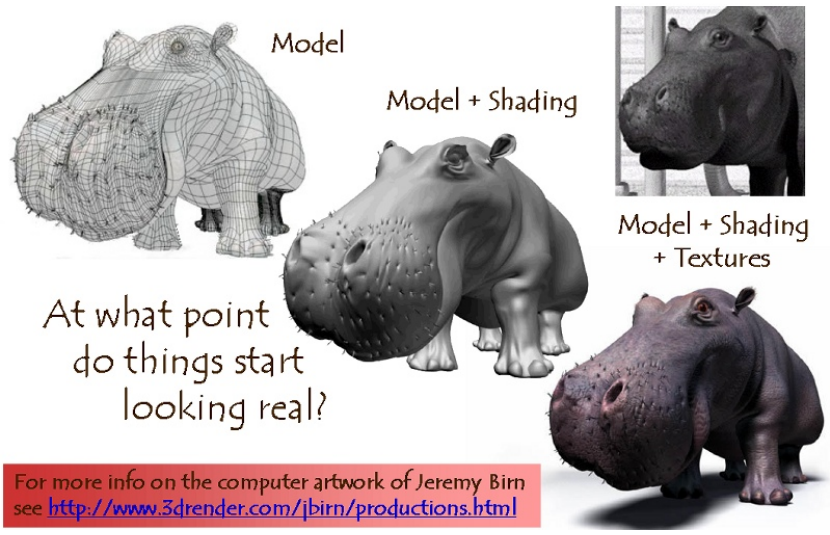
MANCHESTER 1824

Textures

34

MANCHESTER 1824

Modelling and rendering

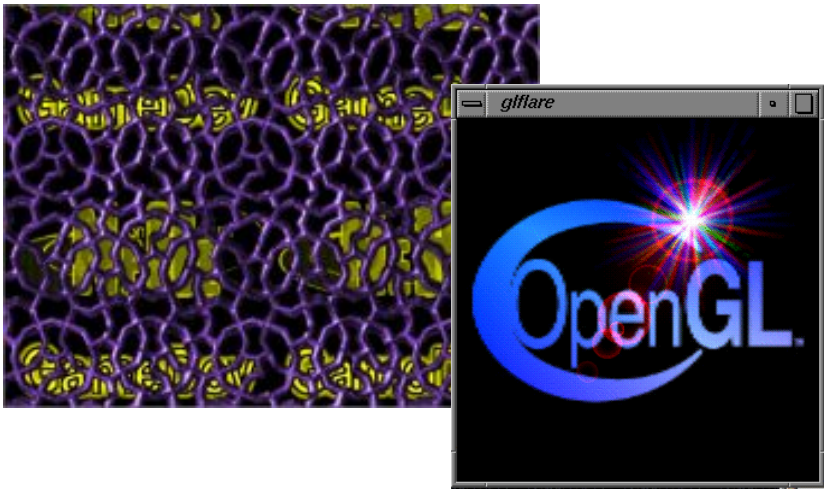


For more info on the computer artwork of Jeremy Birn see http://www.3drender.com/jbirn/productions.html

35

MANCHESTER 1824

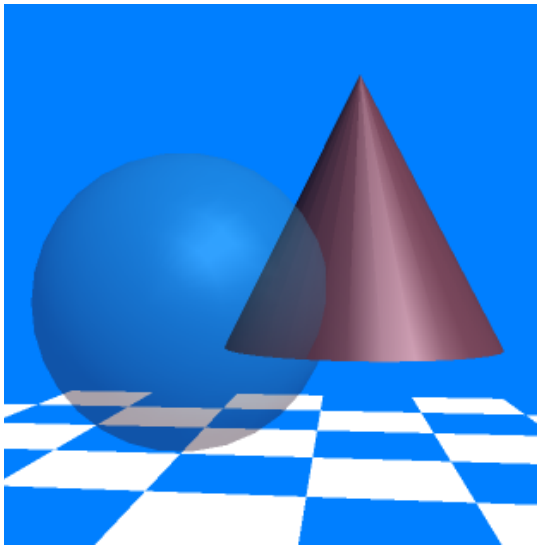
Image compositing



36

MANCHESTER 1824

Alpha-blending for transparency

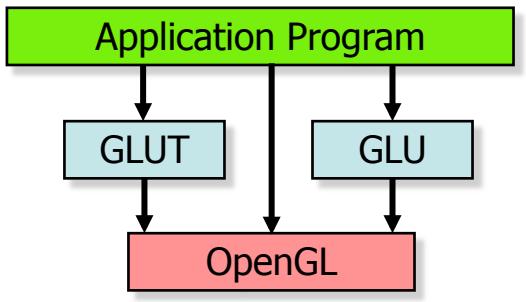


37

MANCHESTER 1824

Support libraries

- OpenGL supports low-level rendering only
- for convenience, two support libraries were developed: GLU and GLUT
- they sit "on top of" OpenGL



38

MANCHESTER 1824

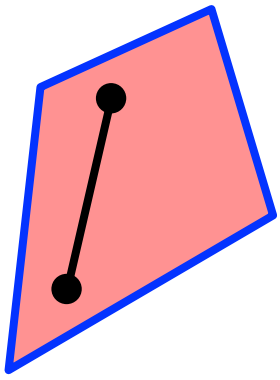
GLU

- GL Utility Library (GLU)
- provides functions which "wrap up" lower-level OpenGL graphics
  - Curves, surfaces, cylinder, disc, quadrics...
- Utility functions
  - Viewing
  - Textures
  - Tessellation
- Command reference:
  - http://www.opengl.org/sdk/docs/man/

39

MANCHESTER 1824

OpenGL needs convex polygons

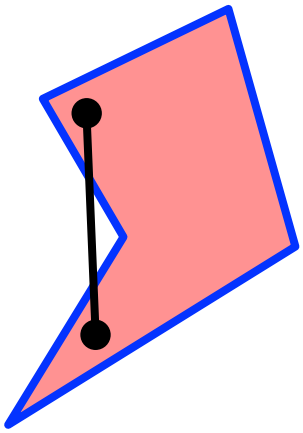


Convex polygon

How to decide if a polygon is convex or not?

Q: Is it possible to find 2 points inside the polygon such that the line between them goes outside the polygon?

A: If it IS NOT possible, the polygon is convex.



Non-Convex polygon
("concave" polygon)

40

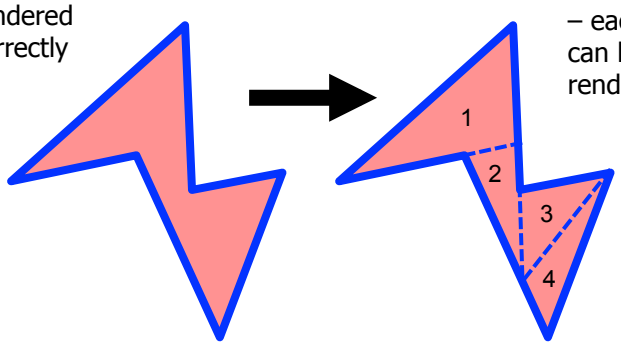
MANCHESTER 1824

Tessellation

- GLU provides functions to tessellate polygons

BEFORE: This **concave** polygon can't be rendered correctly

AFTER: A set of **convex** polygons – each of which can be correctly rendered

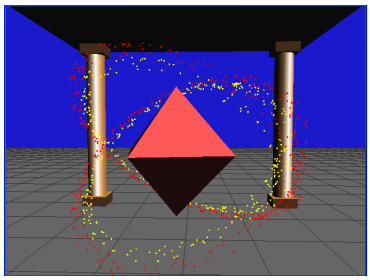


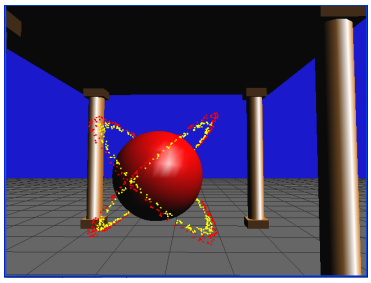
41

MANCHESTER 1824

GLUT

- GL Utility Toolkit (GLUT)
- Interaction
  - interaction (mouse and keyboard)
  - simple menu system
- Primitives
  - Sphere, torus, cone, cube
  - [Tetra | Octo | Dodeca | Icosa] hedron
  - Teapot
- Command reference:
  - Link to PDF on course webpage

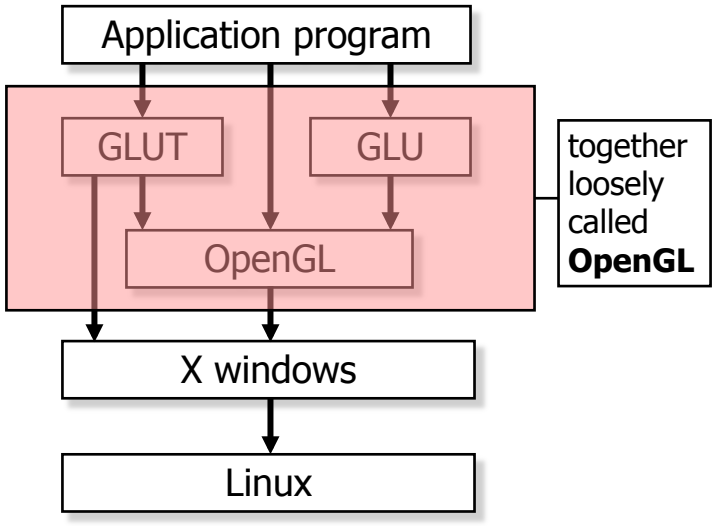




42

MANCHESTER 1824

OpenGL on Linux



43

MANCHESTER 1824

What do do next

1. OpenGL manual – go through Tutorial in Chapters 1-6.
2. Do Coursework 1

44