Graphics 2

06-02408

Level 2
10 credits in Semester 2

Professor Ela Claridge

Aims

- Further develop the concepts and terminology of computer graphics
- Develop understanding of key representations and techniques of computer graphics
- Develop skills in applying computer graphics techniques to construction and viewing problems

On completion of this module, you should be able to

- Design wire-frame representations of 3-dimensional objects
- Define matrices for 3-dimensional transformations
- Explain and design algorithms for viewing and projection of 3-dimensional objects using transformation matrices
- Apply the relevant concepts of linear algebra and geometry to the design of computer graphics algorithms (e.g. vector and matrix operations and trigonometry)
- Explain and design raster conversion algorithms
- Explain image representations and colour models

theguardian

"There is particular criticism of specialised video games and effects courses. In 2009, just 12% of graduates from video games courses found jobs in the sector within six months of graduating. Employers in the games industry say graduates of these courses have poor technical skills in areas such as maths and programming, and lack management skills."

theguardian

Hope, managing director of the visual effects firm Double Negative, said: "In Harry Potter [and the Half Blood Prince], the opening sequence has Death Eaters flying across the river Thames, destroying the [Millennium] bridge between St Paul's and the Tate Modern.

"The way you create that is people who understand computational fluid dynamics, they know how water moves. They take the physics that's used in modelling rivers and the flow of water and apply that in our world. People doing it need an artistic sensibility as well. An understanding of maths and science is fundamental to many of the disciplines in our industry."

Teaching and learning

- "What" and "Why" parts: topic introduction, theoretical underpinnings of the methods used in the practical work
- "How" part: Practical work using Matlab or any software of your choice (e.g. Povray, Java 3D, DirectX, etc) (unassessed)

Syllabus

- Object construction
  - Vertex and surface tables
  - Sweep functions
  - Height maps

•Coordinate systems
•Vectors and matrices + operations
•3D transformations (T,S,R)
•Delunay triangulation

- Rendering
  - Surface colour
  - Lights
  - Reflectance
  - Shading
  - Simple texture mapping
  - Lines and circles

•Light, spectra and colour
•Colour spaces
•Normal vectors
•Physics of light reflectance
  •Cosine law, Snell law
  •Surface properties
•Interpolation

Syllabus (cont.)

- Viewing and projection
  - Camera positioning
  - View definition
  - View projection
  - Hidden surface removal

•Transformation matrices
•Combined transformations
  •Matrix multiplication
•Transformations of coordinate systems

- Animation
  - Object movement
  - Camera movement
  - Path definition
  - Making movies

•Transformations (as above)
•Lines
  •Parametric equations
  •Bresenham's algorithms
•2D and 3D splines
•In-betweening
•Double buffering
•Hidden surface removal

Syllabus (cont.)

- Digital images
  - Handling colour
  - Image enhancement

•Further colour spaces
•Colour conversion
•Colour map manipulation
•Image smoothing and sharpening

- Overview of selected advanced topics
  - Ray tracing
  - Particle animation
  - Morphing

Teaching materials

- Books
- Handouts
- On-line exercises and solutions

Recommended Books

- Computer Graphics, Hearn D & Baker M, 1997.
- 3D Computer Graphics, Watt, A, 2000.

Practical work option: Matlab

- Scientific programming environment
- Interpreted language
- Support for 2D and 3D graphics
- Available for Windows, Mac, Linux

Matlab

- Tutorials
- http://www.cyclismo.org/tutorial/matlab/
  - Work through the tutorial should take you 2-3 hours.
- Matlab Help
  - Have a look at the **"**Programming" and "Graphics" sections

Vectors

- Colours, Positions and Directions are represented using vectors e.g.

```
– rgb [1, 1, 1]
– translate [0, 0, 1]
```

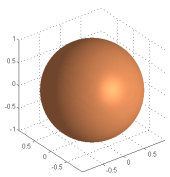
Simple Objects

- A few basic shapes are provided, e.g. a unit sphere is simply generated by

```
S =sphere;
```

- Material properties can be included within the object description e.g.

```
S = sphere;
material shiny;
```



Camera Positioning

The following is an example of a simple set-up :

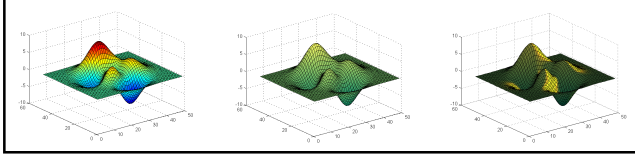
```
camproj('perspective');
campos([xp,yp,zp]);
camup(up_vector);
camtarget([xp;yp;zp]+target_vector);
drawnow
```

There are other options you can use for modelling a camera. We will discuss camera modelling in more detail in later lectures

Light Source

- For any object to be rendered visible, we require at least one light source or emissive object

```
h = surf(peaks);
colormap(spring);
set(h,'FaceLighting','phong',
'FaceColor','interp','AmbientStrength',0.5)
light('Position',[1 0 0],
'Style','infinite','Color',[1 1 0]);
```



Assessment

- 1.5 hr examination (100%)
- The role of the exercises