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 basic raster conversion algorithms
- Explain image representations and colour models

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 Povray (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

•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

•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.)

- Creating complex objects
  - Hierarchical grouping
  - Assigning properties
- Digital images
  - Handling colour
  - Image enhancement

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

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.
- http://web.mit.edu/6.094/www/lecnotes/lec1.ppt
  - Ignore first five pages which have information relevant to the MIT course
- Matlab Help
  - Have a look at the **"**Programming" and "Graphics" sections

Assessment

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

Teaching materials

- Books
- Handouts
- On-line exercises and solutions

Recommended Books

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

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

Comments

- Comments can be included like this

```
figure;
  hold on %this is a single line comment
%{
% the whole section commented out
for i=1:N
  plot(X,Y,'r');
end
%}
```