# COMP 261 Lecture 13

3D Graphics 1 of 2

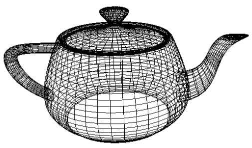
## 3D graphics

- Different business from 2D images!

Model of 3d objects
- shape
- surface properties
- material/mass/..
- movement/animation
- light sources
- …

Rendering →

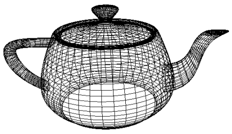
Image to display



## Assignment 3

- Outline
  - Polygons
  - Move them
  - Lighting
  - Hiding
  - Scan conversion ("filling in…")
- Difficult bits:
  - Scan conversion
  - Overall program structure
- Approximate effort: maybe 600 lines of code

## Modelling shapes

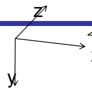


Lots of schemes:

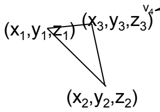
- Polygon meshes
  - vertices, edges, polygon faces
  - Common in computer games because enables fast rendering especially with high end video cards
- Mathematical functions defining surfaces
- Point clouds / particle clouds
- Voxel based (MRI)
  - 3D pixels: full or empty
- CSG Constructive Solid Geometry
  - shapes made out of primitive shapes, scaled, rotated, translated, skewed, added, subtracted, intersected
- …………..

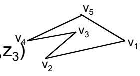
## Polygons

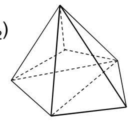
Coordinate system: Right-hand rule




- Represent polygon faces:
  - vertices
    - $v_1 = (x_1,y_1,z_1)$,
    - $v_2 = (x_2,y_2,z_2)$,
    - …
    - $v_3 = (x_n,y_n,z_n)$
    - In order
  - flat or twisted?
    - not an issue if a triangle!!
  - which is the outside?
    - facing you, if the vertices are anti-clockwise
    - surface **normal** is a unit vector facing outwards
    - $= (v_2-v_1) \times (v_3-v_2) \ / \ |(v_2-v_1) \times (v_3-v_2)|$



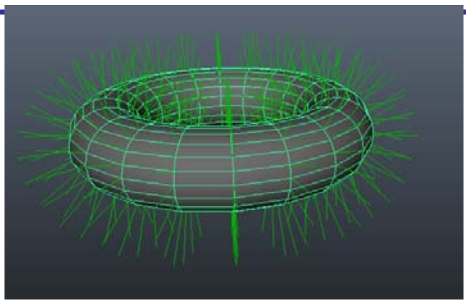



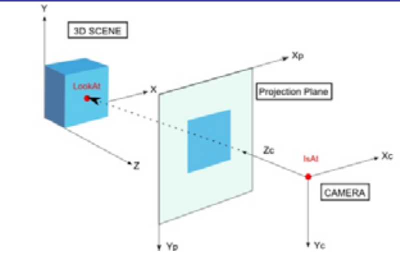
cross product

direction of normal: right-hand rule!

## Normal vector

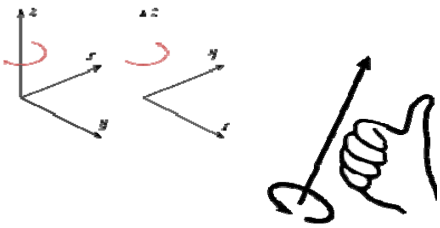


## Forward and backward facing polygons




Pda-fx.net

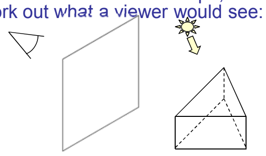
## Right (or left) hand rule

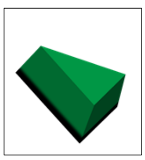


wik
i

## Polygon Rendering

- Given the model of the shape,
  Work out what a viewer would see:

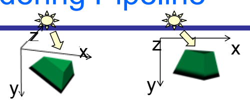




**Issues**

- representation of polygons
- changing to the viewer's perspective: rotation, translation, perspective
- hidden parts and objects obscuring others
- illumination and surface effects
- [shadows and reflections from other objects]

## Simple Z-buffer Rendering Pipeline



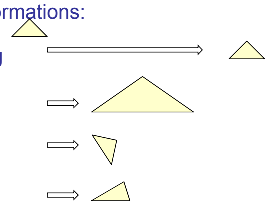

**input**: set of polygons
(position, colour)
viewing direction
direction of light source(s)
size of window.

**output**: an image

**Actions**
- rotate polygons and light source so viewing along z axis
- translate & scale to fit window / clip polygons out of view
- remove any polygons facing away from viewer ($normal_z > 0$)
- for each polygon
  - compute shading
  - work out which image pixels it will affect
  - for each pixel write shading and depth to z-buffer (retains only the shading of the closest surface)
- convert z-buffer to image

## Polygon transformations

Affine transformations:



- Translating
- Scaling
- Rotating
- Shearing

  - linear transformations,
  - preserve colinearity and ratios of distances along a line
  - may NOT preserve angles, lengths, areas
  - Can be represented using vectors & matrices

## Linear transformations

- Translation:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} \Rightarrow \begin{pmatrix} x+\Delta x \\ y+\Delta y \\ z+\Delta z \end{pmatrix}$$

- Scale:

$$\begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{pmatrix} \times \begin{pmatrix} x \\ y \\ z \end{pmatrix} \Rightarrow \begin{pmatrix} x \bullet s_x \\ y \cdot s_y \\ z \cdot s_z \end{pmatrix}$$

matrix multiplication

- Rotation:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix} \times \begin{pmatrix} x \\ y \\ z \end{pmatrix} \Rightarrow \begin{pmatrix} 1x + 0y + 0z \\ 0x + \cos\theta y - \sin\theta z \\ 0x + \sin\theta y + \cos\theta z \end{pmatrix}$$

Problem: different kinds of operations

## Linear transformations

- Translation: $\begin{bmatrix} \\ \\ \end{bmatrix} + \begin{bmatrix} \\ \\ \end{bmatrix} \Rightarrow \begin{bmatrix} \\ \\ \end{bmatrix}$
- Scale: $\begin{pmatrix} 5 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \Rightarrow \begin{bmatrix} 5 \\ 2 \\ 3 \end{bmatrix}$
- Rotation: $\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix} \times \begin{bmatrix} \\ \\ \end{bmatrix} \Rightarrow \begin{pmatrix} \\ \\ \end{pmatrix}$

## Matrix multiplication

- Size checking rules
  - (4 x 3) * (3 x 1) = (4 x 1)
  - Inner dimensions compatible (3,3), and contracted away
  - Outer dimensions retained (4,1)

## Linear transformations

- Translation: $\begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} \Rightarrow \begin{bmatrix} x+\Delta x \\ y+\Delta y \\ z+\Delta z \end{bmatrix}$
- Scale: $\begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{pmatrix} \times \begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow \begin{bmatrix} x \cdot s_x \\ y \cdot s_y \\ z \cdot s_z \end{bmatrix}$

matrix multiplication

- Rotation: $\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix} \times \begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow \begin{pmatrix} 1x + 0y + 0z \\ 0x + \cos\theta y - \sin\theta z \\ 0x + \sin\theta y + \cos\theta z \end{pmatrix}$