Computer Graphics

P. Healy

CS1-08
Computer Science Bldg.
tel: 202727
patrick.healy@ul.ie

Spring 2021–2022

Outline

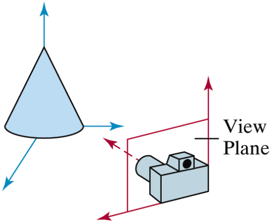
Outline

3-D vs. 2-D

- Providing realistic 3-D graphics is a more difficult task than the 2-D case
- Some issues such as are common to both
- However, realistic 3-D scenes must provide to the viewer **perspective** by projecting the scene on to a 2-D surface...
- ...this involves identifying the **visible** parts of a scene
- This is related to the **camera position** we choose
- For a realistic display: **lighting** effects and **surface** characteristics

Camera Positioning

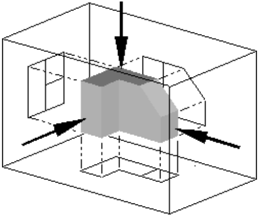
- The first step in obtaining a 3-D display: set up a coordinate reference for viewing (camera) position
- This defines for a **view plane**

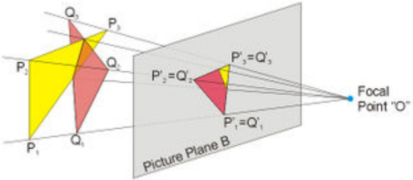


- Objects are then projected (how??) on to view plane
- View of scene can be generated in wire-frame (outline) form or more realistically using sophisticated techniques

Projections

- Two ways to project scene on to view plane
- ______ (left) or ______ (right)





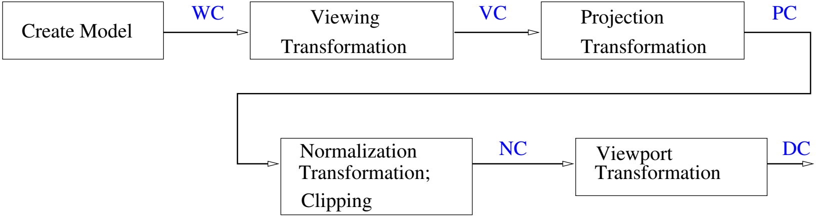
- Loss of detail in perspective projection can be helped by ______, which draws points closer to the camera with greater intensity

Surface Rendering

- Added realism can be attained by rendering object surfaces using lighting conditions in the scene and surface characteristics
- We specify colour and location of the light source and background illumination effects
- properties can also be modelled such as transparent, opaque, and reflectivity (rough vs. smooth)
- Can even get down to setting parameters that give bumpy appearance of an orange as opposed to wood

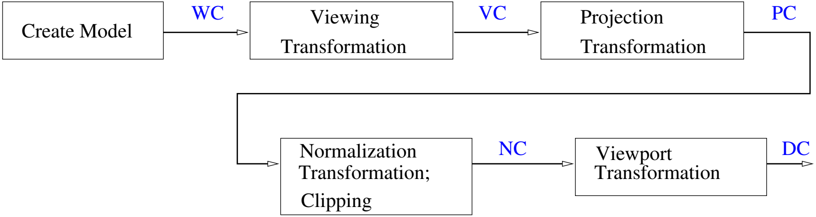
Outline

Viewing Pipeline



- coordinates are also known as **eye** coordinates or **camera** coordinates See Qs 9.001 & 9.011
- Very good overview of OpenGL 3-D viewing pipeline here

Viewing Pipeline

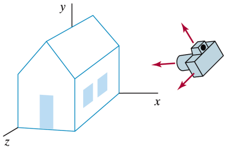


- coordinates are also known as **eye** coordinates or **camera** coordinates See Qs 9.001 & 9.011
- Very good overview of OpenGL 3-D viewing pipeline here

Outline

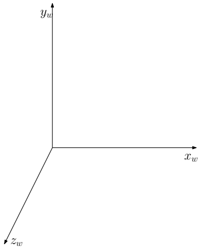
1. Three-Dimensional Viewing – §10
   - Overview of 3-D Viewing Concepts
   - Three-Dimensional Viewing Pipeline
   - Three-Dimensional Viewing Coordinates – §10.3

Viewing Coordinates



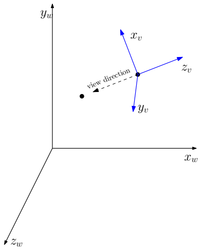
- Similar to 2-D pipeline we need to maintain different coordinate systems for world coords and view coords
- Point p_0 in world coords is position of camera
- If we specify a **view-up vector**, V (the y_v axis), and a **viewing direction** (usually along negative z_v axis) we get...

Viewing Coordinates (contd.)



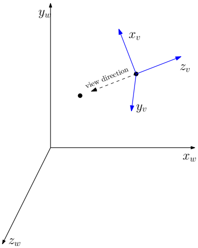
- Note use of right-hand twist rule in both coord systems above and previous slide...
- ...and unrelated positions of two *y*-axes;
- See also §9-6 of *H-B* and wikipedia.

Viewing Coordinates (contd.)



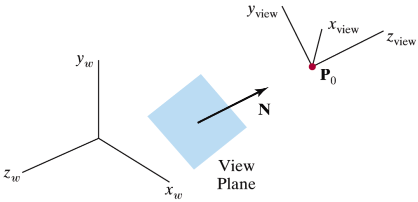
- Note use of right-hand twist rule in both coord systems above and previous slide...
- ...and unrelated positions of two y-axes; but don't read : the y-axis does **not** have to point upwards in RHC/LHC!
- See also §9-6 of *H-B* and wikipedia.

Viewing Coordinates (contd.)



- Note use of right-hand twist rule in both coord systems above and previous slide...
- ...and unrelated positions of two *y*-axes;
- See also §9-6 of *H-B* and wikipedia.

Viewing Coordinates (contd.)



- Perpendicular to viewing direction is the **view plane**
- This is always parallel to $x_v - y_v$ plane
- The projection of objects to the view plane determines the view of scene displayed
- N, the **view plane normal**, is a way to specify the viewing direction
- N can be set as the vector between some point of reference, p_{ref}, in scene and p_0: $N = p_0 - p_{\text{ref}}$;
- N points back in our face – against viewing direction, $-N$

Viewing Coordinates (contd.)

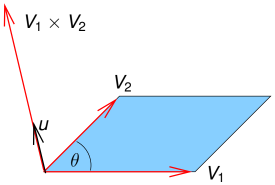
- The usual order of events is
  1. select p_0 and p_{ref} (in world coordinates)
  2. This defines viewing direction and N, the normal to the view plane
  3. V must be perpendicular to N (but this doesn't tie it down yet)
  4. Usually: choose any pseudo-direction V' for view-up vector as long as it's not parallel to N...
  5. ... and *project* this on to view plane
  6. This gives the direction for V
  7. A common choice for V' is the world y axis; that is, vector $(0, 1, 0)$
- We now describe how to achieve above using **cross product** operator

uvn Viewing Coordinates

- Suppose we had vectors that represent the directions of N and V, the equivalents of the $-z_v$ and y_v axes of the viewing coord system
- The third direction is no longer a degree of freedom, but what is it?
- Also, we want to know what the "unit" vectors in viewing coordinates
- We make use of the cross product operator of vectors

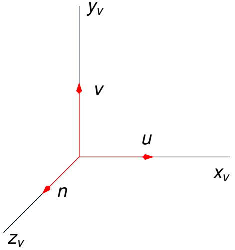
$$V_1 \times V_2 = u|V_1||V_2|\sin\theta$$

where u is a vector of unit length perpendicular to the plane that V_1 and V_2 lie in (see p. 762 of *HB* for more details)



uvn Viewing Coordinates (contd.)

- With N and V' given to us (as "inputs") we firstly **normalize** them to get their unit-length equivalents



uvn Viewing Coordinates (contd.)

- With N and V' given to us (as "inputs") we firstly **normalize** them to get their unit-length equivalents

$$
\begin{aligned}
n &= \frac{N}{|N|} && = (n_x, n_y, n_z) \\
v &= \frac{V'}{|V'|} && = (v_x, v_y, v_z) \\
u &= v \times n && = (u_x, u_y, u_z)
\end{aligned}
$$

Right?

uvn Viewing Coordinates (contd.)

- With N and V' given to us (as "inputs") we firstly **normalize** them to get their unit-length equivalents

$$
\begin{aligned}
n &= \frac{N}{|N|} &= (n_x, n_y, n_z) \\
v &= \frac{V'}{|V'|} &= (v_x, v_y, v_z) \\
u &= v \times n &= (u_x, u_y, u_z)
\end{aligned}
$$

WRONG!!

Remember V' is only a rough, "sort of this way", direction and, in general, won't be on the plane perp. to N, the view plane;

solution: firstly find u, the vector that is perp. to plane that contains V' (this is perp.) and N, and then use n and u to find v

uvn Viewing Coordinates (contd.)

- With N and V' given to us (as "inputs") we firstly **normalize** them to get their unit-length equivalents

$$
\begin{aligned}
n &= \frac{N}{|N|} && = (n_x, n_y, n_z) \\
u &= \frac{V' \times n}{|V' \times n|} && = (u_x, u_y, u_z) \\
v &= n \times u && = (v_x, v_y, v_z)
\end{aligned}
$$