COMP 3069
Computer Graphics

Lecture 5:
Viewing

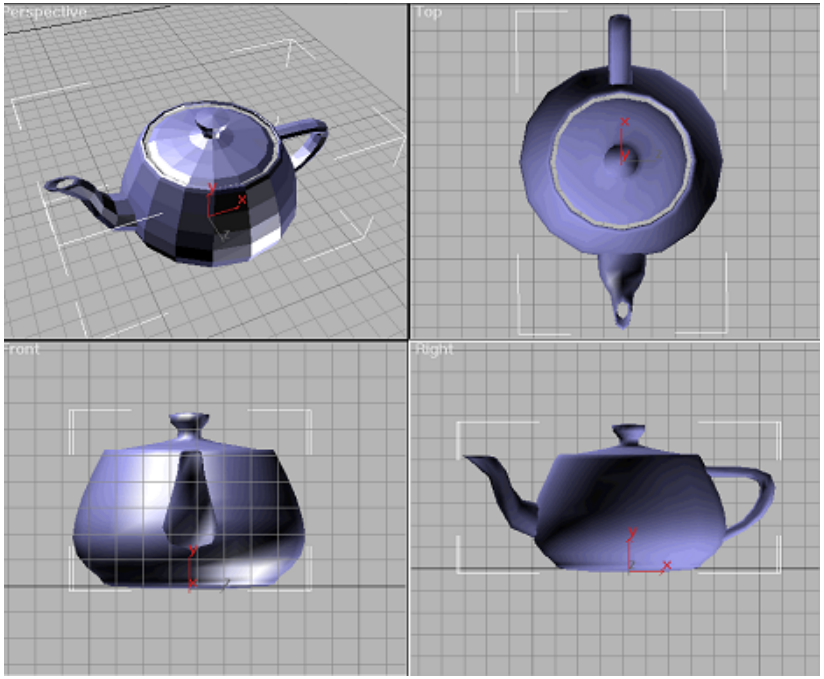
Autumn 2018

The University of Nottingham

UNITED KINGDOM · CHINA · MALAYSIA

Transformation from World to View Space

- Recap last week (animation)
- Graphics coordinate systems
- Virtual camera setup
- World to View Transformation
- An example transformation
- A demo of camera setup
- Coursework briefing
- Coursework examples
- Framework for the coursework



Graphics Coordinate Systems

- The geometry of objects is expressed in **world coordinate system (or world space)**.

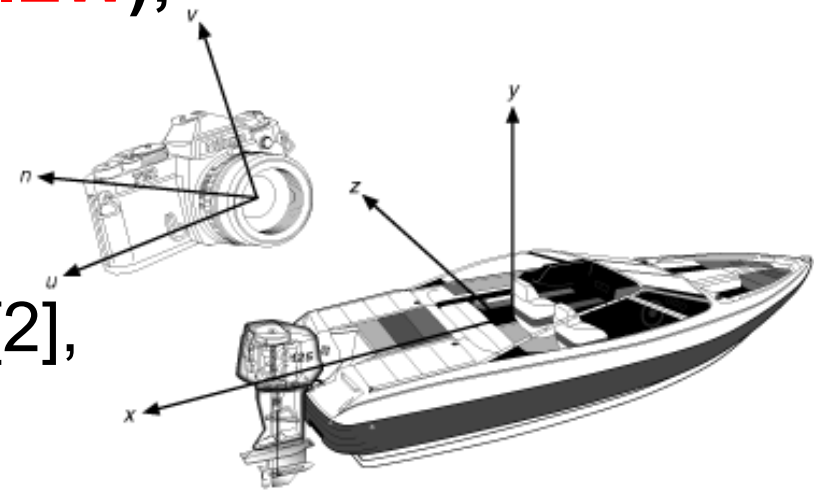
- OpenGL by default creates an object centered at the coordinate origin of the world space, but you can specify the position of an object by giving exact x, y, z values for vertices,

  E.g., glVertex(10, 20, 30)

- Objects are then transformed to **view coordinate system (or view space or camera space)** so that they can be projected to the 2D image plane, analogous to taking a photograph with a camera.

OpenGL Camera Setup – World to View Matrix

```
void Camera::SetUpCamera()
{
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
gluLookAt(
    eye[0], eye[1], eye[2],
    lookAt[0], lookAt[1], lookAt[2],
    up[0], up[1], up[2]);
}
```

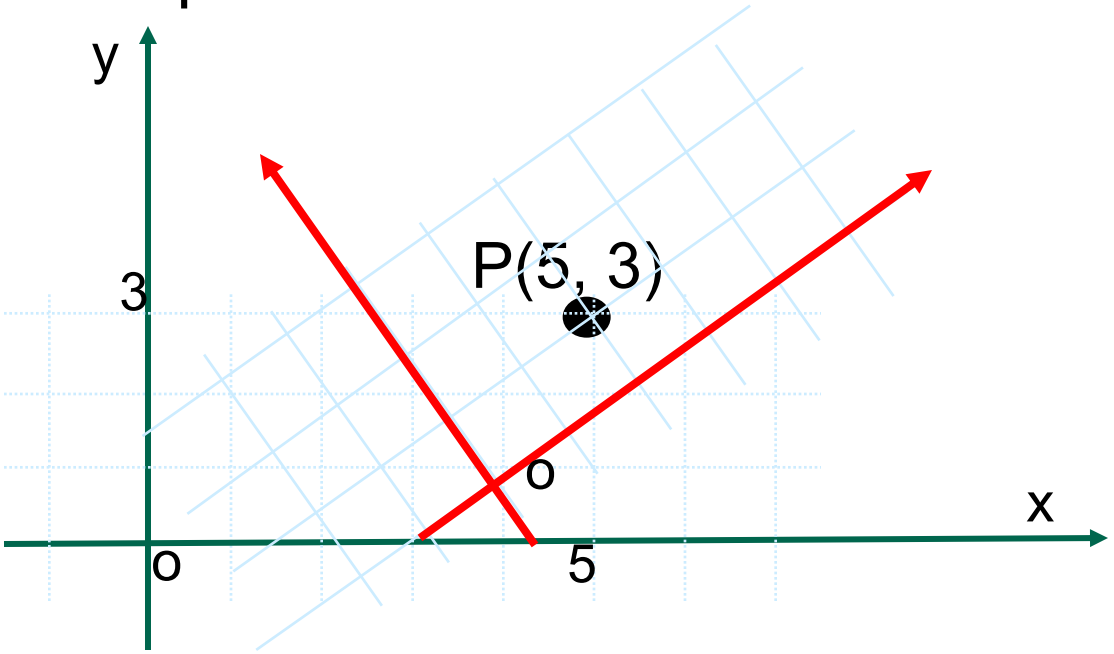


- eye - camera position in world coordinate system
- lookAt - camera viewing direction = lookAt - Eye
- up - camera up direction

Changing Coordinate Systems

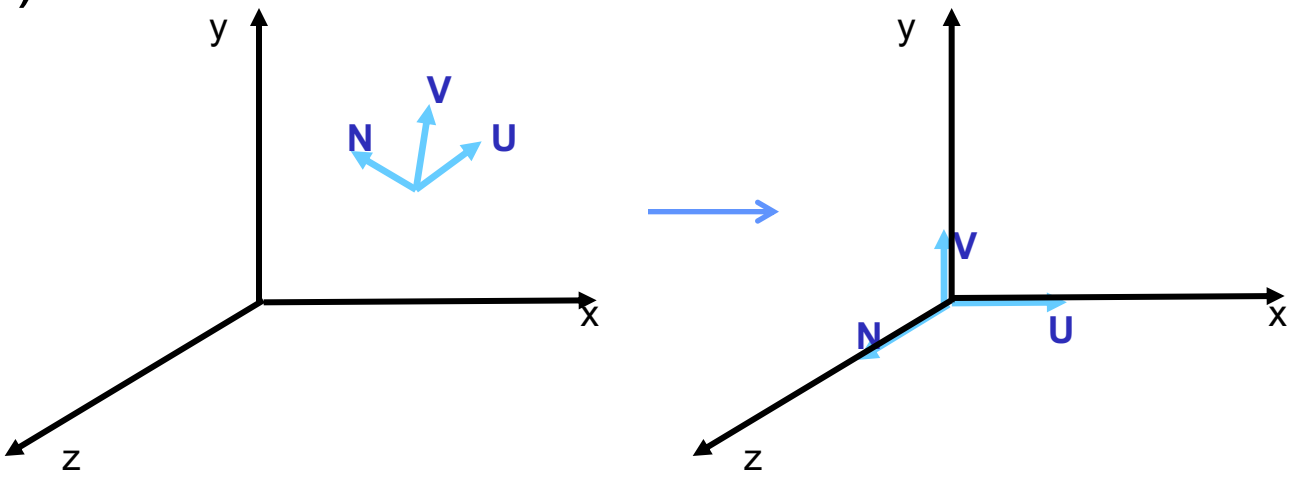
- Suppose the world coordinate axes are in **green** and view (camera) coordinate axes are in **red**

- The coordinate of a point (the black dot below) in world space is (5, 3). What's the point's coordinate in view space?



Given coordinates P in world space, how do we calculate coordinates P' in view space?

- This can be done by a transformation that aligns view coordinate axes with world coordinate axes:
  - T to translate the view coordinate origin to world coordinate origin, then
  - R to rotate to align view coordinate axes u, v, n with x, y, z of world space
  - Then (RT) P = P'



Rotation Matrix to Align View and World Coordinate Axes

- World coordinate axes are
$x = (x_1, x_2, x_3)$, $y = (y_1, y_2, y_3)$, and
$z = (z_1, z_2, z_3)$

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- View coordinate axes are
$u = (u_x, u_y, u_z)$, $v = (v_x, v_y, v_z)$, and
$n = (n_x, n_y, n_z)$,

- We need to find a rotation matrix R to satisfy the following 3 simultaneous equations

$x = R*u$, $y = R*v$, $z = R*n$ $\longrightarrow$ $R = \begin{bmatrix} u_x & u_y & u_z & 0 \\ v_x & v_y & v_z & 0 \\ n_x & n_y & n_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

- Solving the 3 equations gives us

World to View Transformation – How?

- So we want to find a rotation matrix *R*, such that:

$$RU = R\begin{pmatrix} u_x \\ u_y \\ u_z \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad RV = R\begin{pmatrix} v_x \\ v_y \\ v_z \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \qquad RN = R\begin{pmatrix} n_x \\ n_y \\ n_z \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

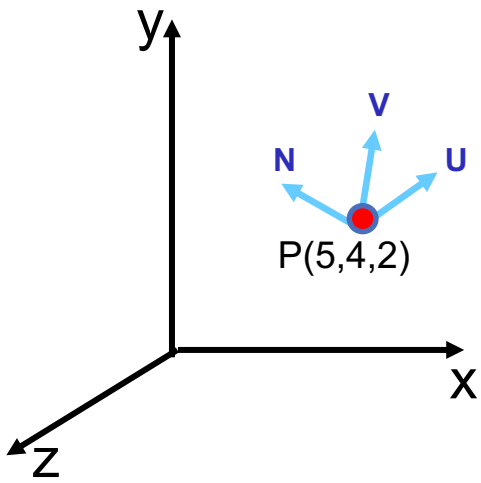
- Each green box represents 3 equations, that is, we get 3 equations from R * U = X, 3 equations from R * V = Y, and 3 equations from R * N = Z.
- Equation solving omitted …

World to View Transformation

- So if coordinate of a point in world space is P, and in view space is Q, then

$$P' = R * T * P \text{ , and}$$

$$T^{-1} * R^{-1} * P' = P$$



where

$$R = \begin{bmatrix} u_x & u_y & u_z & 0 \\ v_x & v_y & v_z & 0 \\ n_x & n_y & n_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \quad T = \begin{pmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

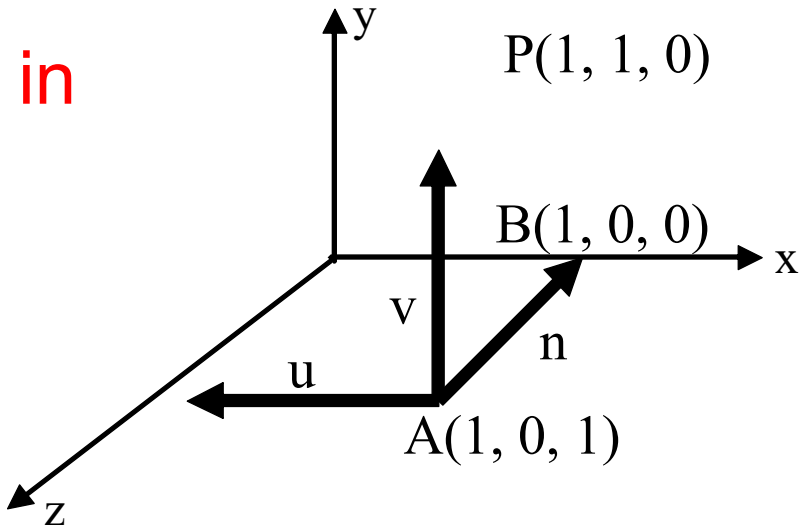
World to View Transformation Exercise

- A viewing coordinate system is set up with the ***eye*** at A(1,0,1), ***lookat*** at B(1,0,0), and 'up' vector is the positive y direction.
- P is a point in world coordinate system. Derive an expression that gives P's coordinates in the view coordinate system.
- Solution will be given in the lecture

u = -x

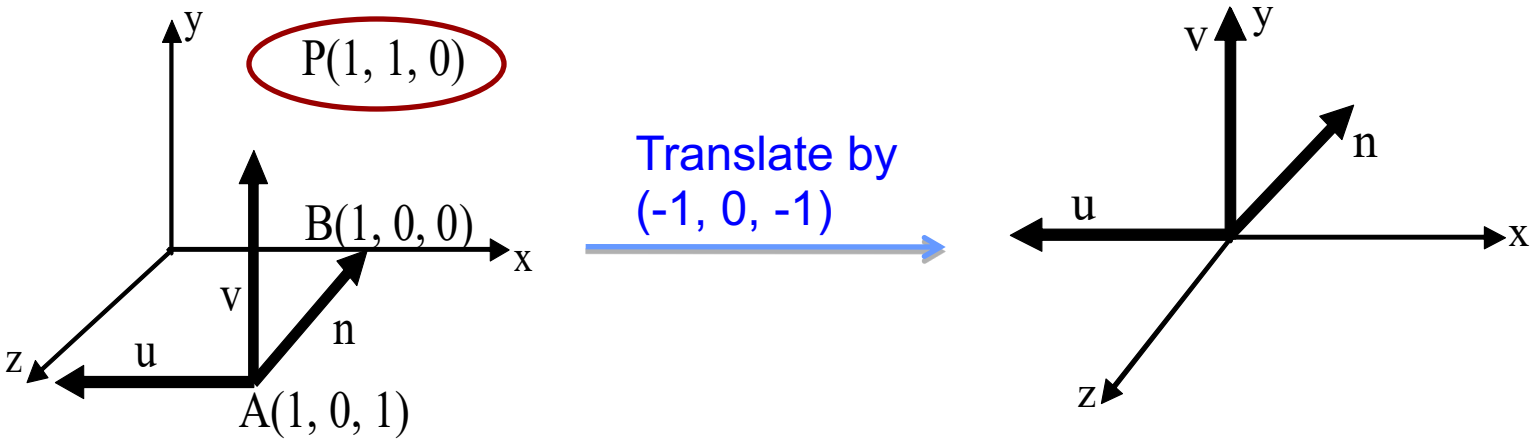
v = y

n = -z



World to View Exercise

First translate the camera center to the origin



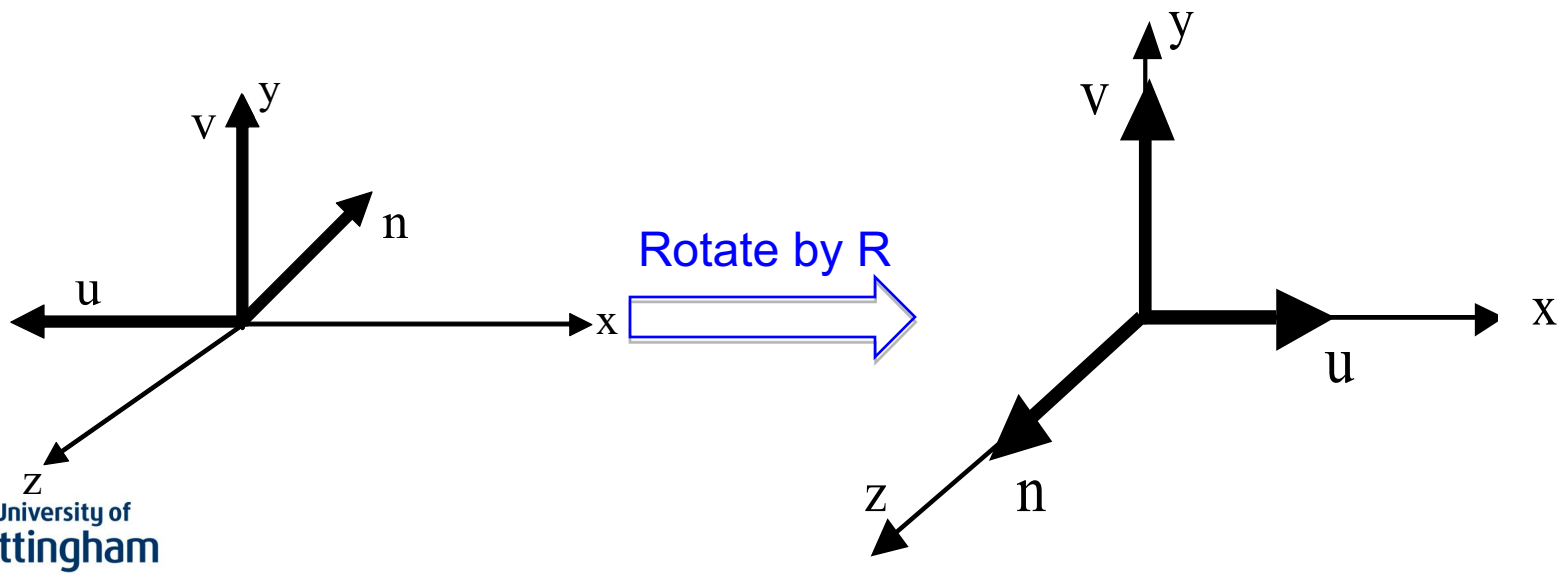
Translation matrix: $T = \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

World to View Exercise

Then rotate to align the axes:

As u = (-1, 0, 0), so
v = (0, 1, 0),
n = (0, 0, -1)

$$R = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



World to View Exercise

$$\underset{R}{\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}} \underset{T}{\begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix}} \underset{P}{\begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \end{pmatrix}}$$

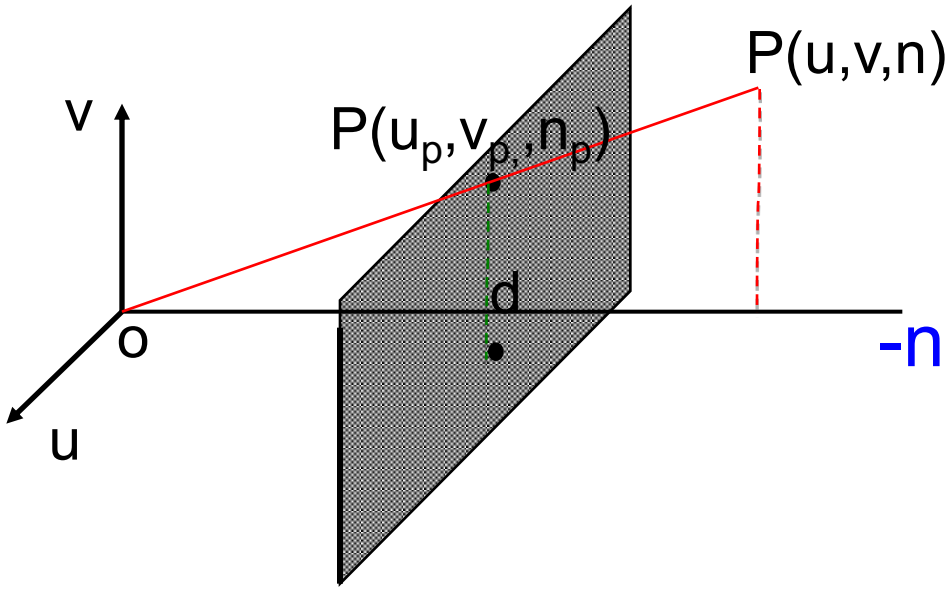
$$= \begin{bmatrix} -1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

So a point **P** in world space, can be transformed to view space by a **translation** (**T**) and then by a **rotation** (**R**):

Q = R T P

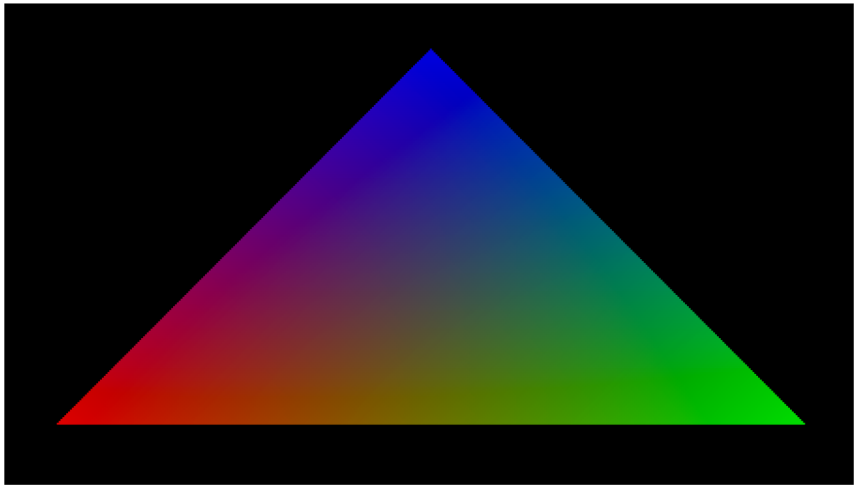
Projecting 3D objects onto Camera Image Plane (View Plane) – Next Week

- Now our objects / 3D world is in the view/camera coordinate system, ready to be projected onto 2D camera image plane or view plane.



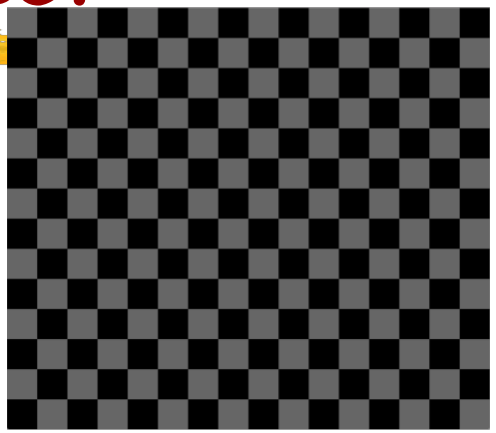
The Viewing Demo – Creating a Triangle Object

```
glBegin(GL_TRIANGLES);
glColor3ub(255, 0, 0);
glVertex3d( -40, -00, 0.0 );
glColor3f(0, 1.0, 0);
glVertex3d(  40, -00, 0.0 );
glColor3ub(0, 0, 255);
glVertex3d( 0.0,  40, 0.0 );
glEnd();
```



How do you set up the camera to view the triangle?

The Viewing Demo – Creating a Floor Object:



```
for(int i = -10; i <= 10; i ++) {
    for(int j = -10; j <= 10; j ++) {
...
glBegin(GL_QUADS);
glVertex3f( S*(float) i, -10.0f, S*(float) j);
glVertex3f( S*(float) i, -10.0f, S*(float) j + S);
glVertex3f( S*(float) i + S, -10.0f, S*(float) j + S);
glVertex3f( S * (float) i + S, -10.0f, S * (float) j);
glEnd();
}
}
```

S is short for SIZE

How do you set up the camera to view the floor?

Coursework Briefing

- It is time to think about your coursework
- As mentioned in the introduction lecture, the coursework is about implementing a graphics application, of your choice
- Think about what you'd like to do, e.g.,
  - What you would like to model and/or animate
  - How you could make use of what you have learnt for your coursework
  - Write down your ideas and discuss your ideas with your peers, tutors, or lecturer

Figure 4: Young winter tree example

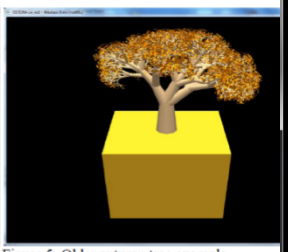


Figure 5: Older autumn tree example