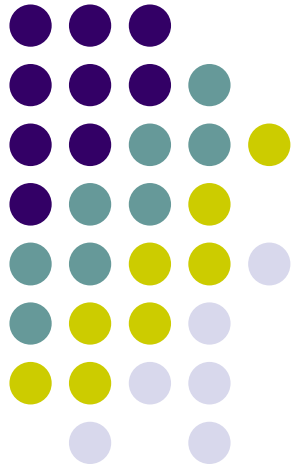
Computer Graphics (CS 543)
Lecture 6c: Introduction to Projection

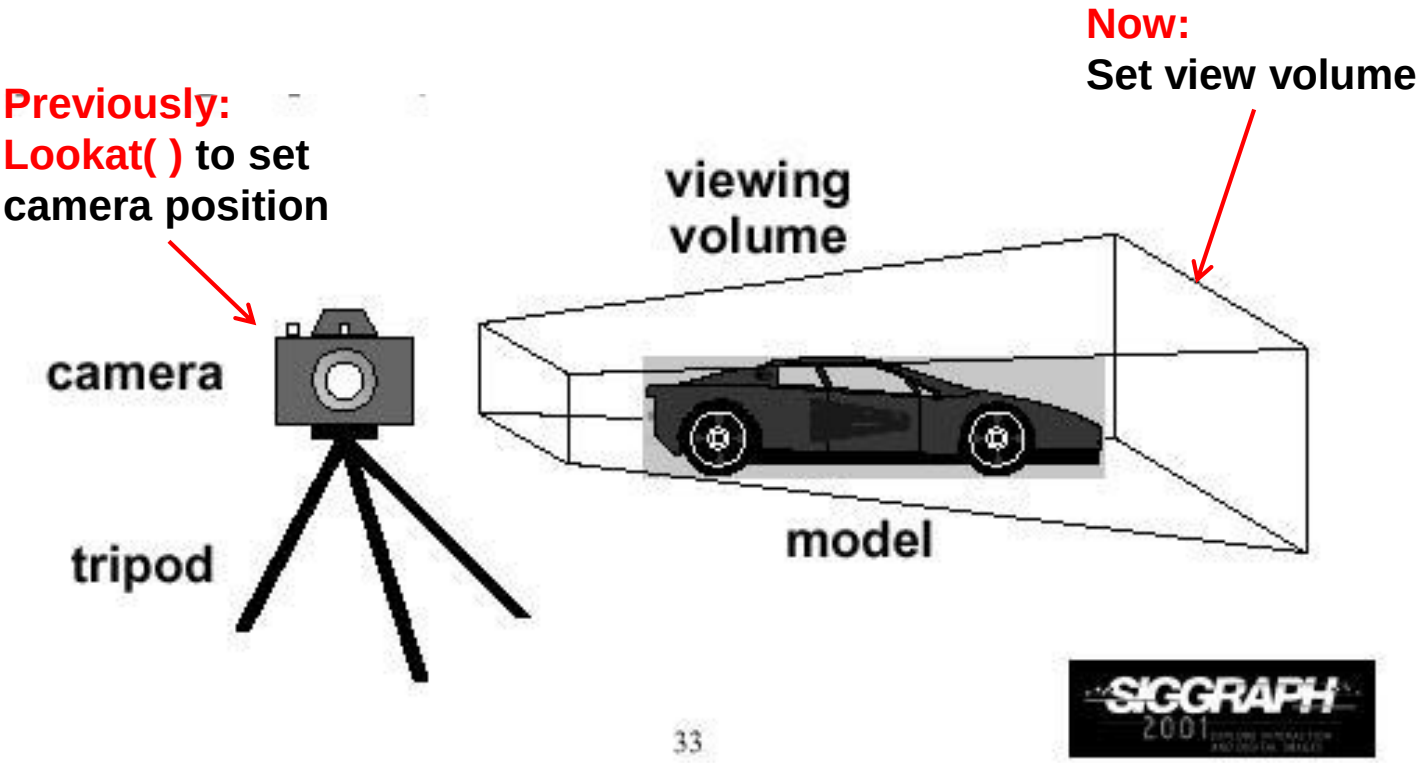
Prof Emmanuel Agu

Computer Science Dept.

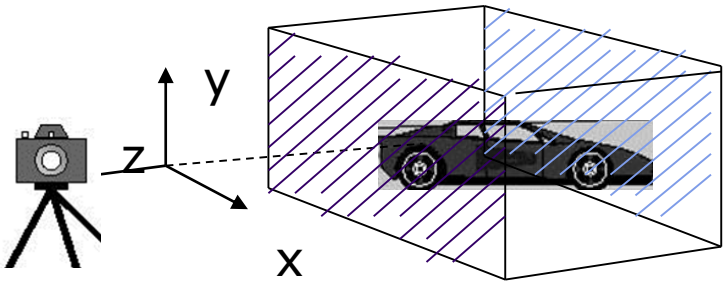
Worcester Polytechnic Institute (WPI)



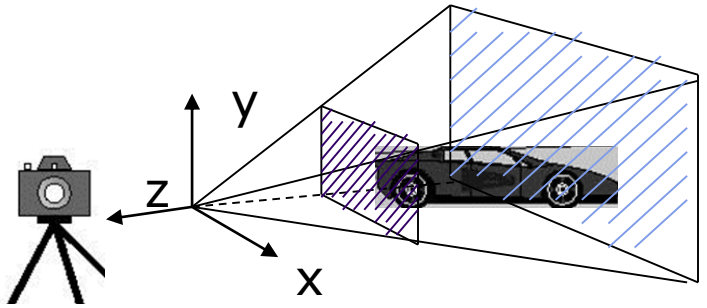
Recall: 3D Viewing and View Volume



Recall: Different View Volume Shapes



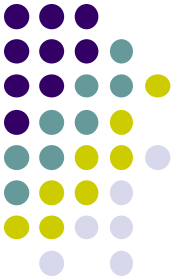
Orthogonal view volume
(no foreshortening)



Perspective view volume
(exhibits foreshortening)

- Different view volume => different look
- **Foreshortening?** Near objects bigger

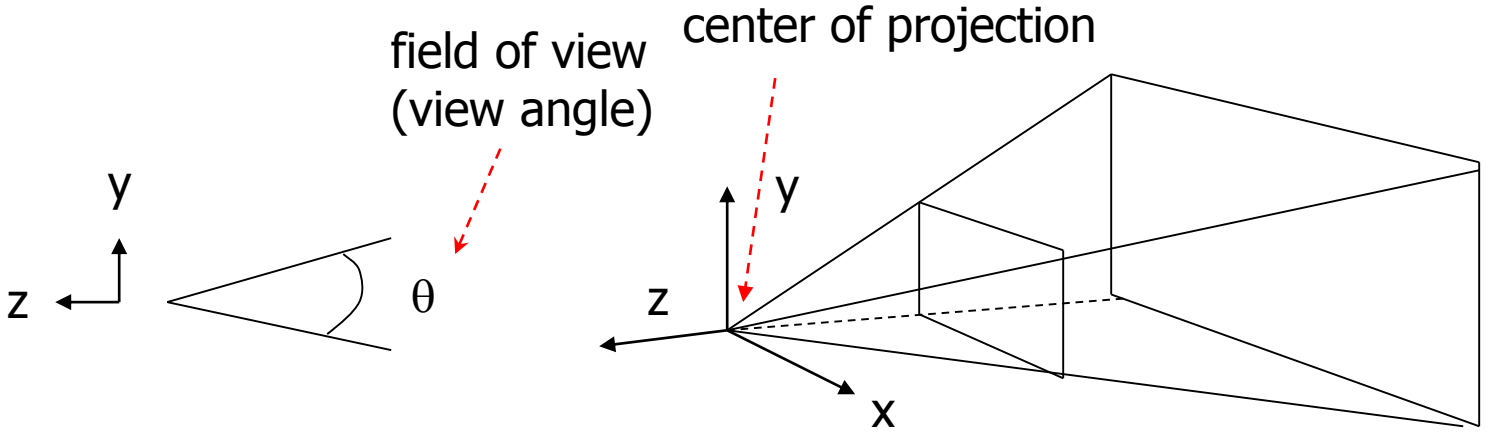
View Volume Parameters



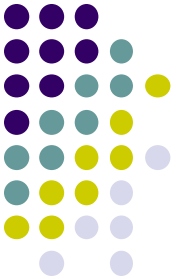
- Need to set view volume parameters
  - **Projection type:** perspective, orthographic, etc.
  - Field of view and aspect ratio
  - Near and far clipping planes

Field of View

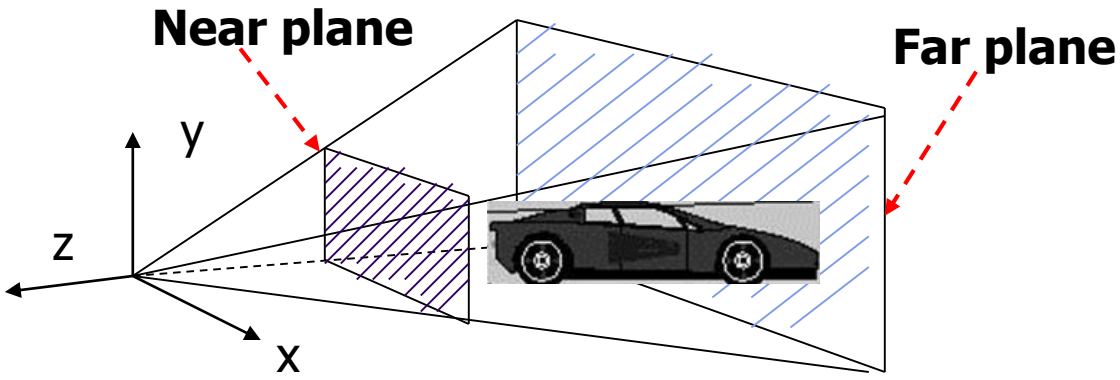
- View volume parameter
- Determines how much of world in picture (vertically)
- Larger field of view = smaller objects drawn



Near and Far Clipping Planes

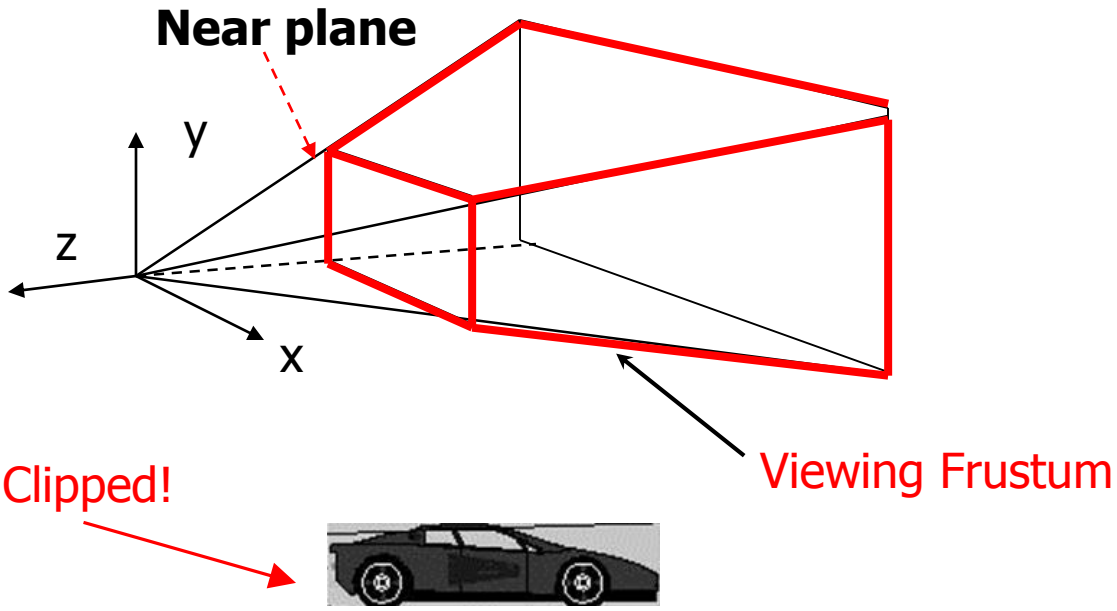


- Only objects between near and far planes drawn

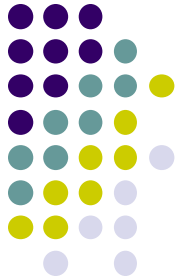


Viewing Frustrum

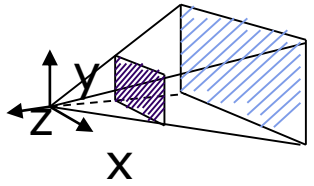
- Near plane + far plane + field of view = **Viewing Frustum**
- Objects outside the frustum are clipped



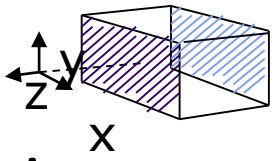
Setting up View Volume/Projection Type



- Previous OpenGL projection commands **deprecated**!!
  - Perspective view volume/projection:
    - **gluPerspective**(fovy, aspect, near, far) **or**
    - **glFrustum**(left, right, bottom, top, near, far)



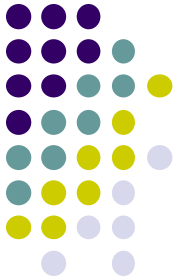
  - Orthographic:
    - **glOrtho**(left, right, bottom, top, near, far)



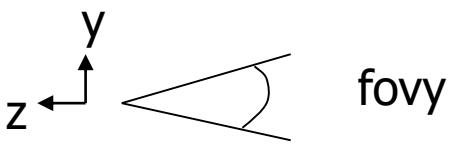
- Useful functions, so we implement similar in `mat.h`:
  - **Perspective**(fovy, aspect, near, far) **or**
  - **Frustum**(left, right, bottom, top, near, far)
  - **Ortho**(left, right, bottom, top, near, far)

What are these arguments? Next!

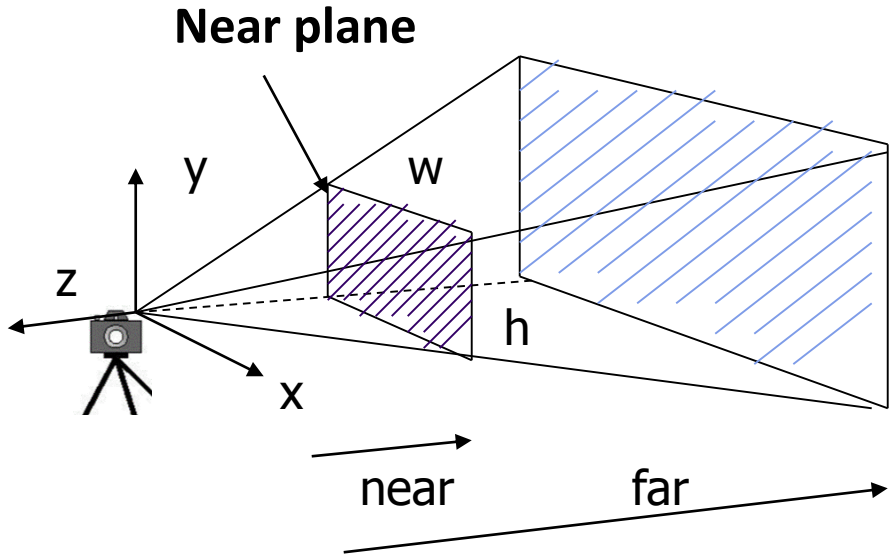
Perspective(fovy, aspect, near, far)



- Aspect ratio (of near plane) used to calculate window width

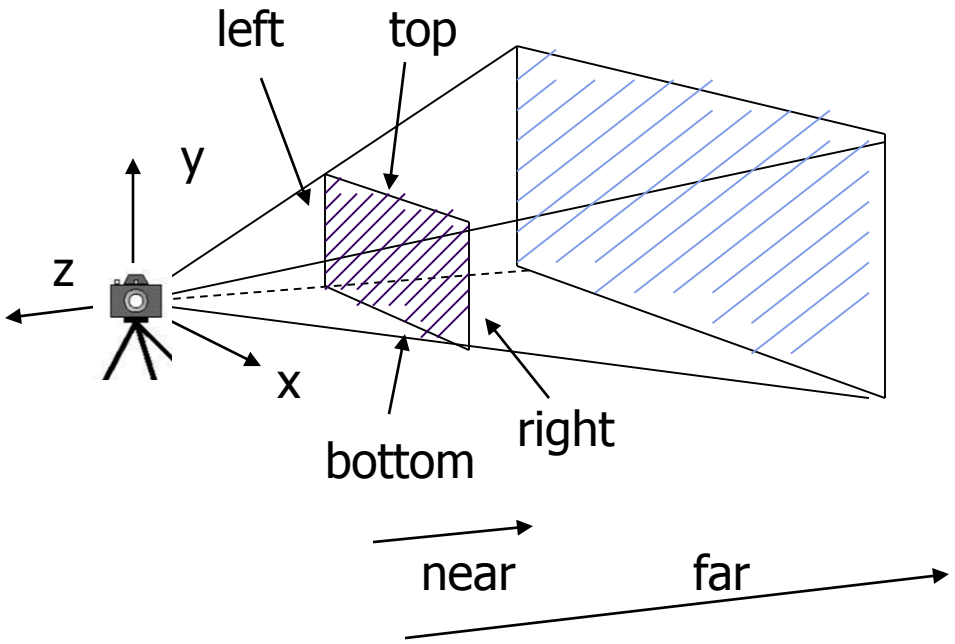


Aspect = w / h



Frustum(left, right, bottom, top, near, far)

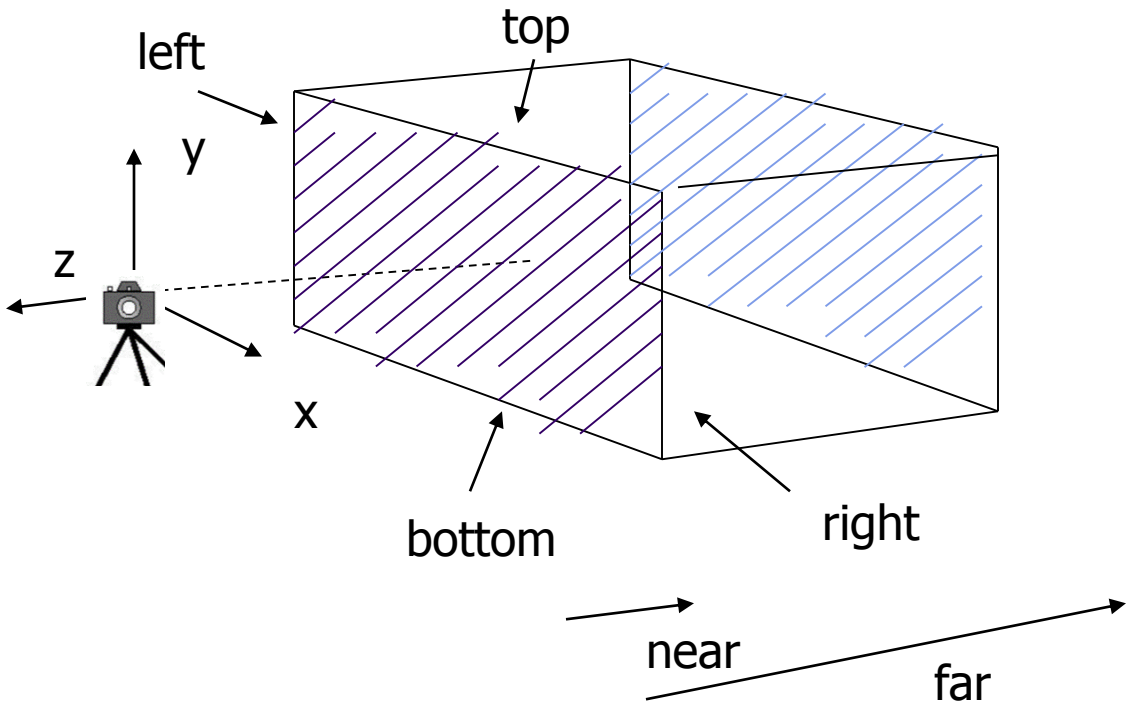
- Can use **Frustrum( )** in place of **Perspective**()
- Same view volume **shape**, different **arguments**



`near` and `far` measured from camera

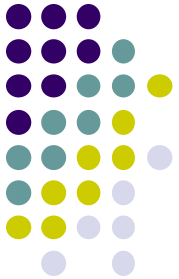
Ortho(left, right, bottom, top, near, far)

- For orthographic projection



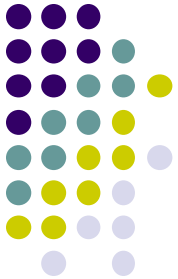
`near` and `far` measured from camera

Demo



- Nate Robbins demo on projection

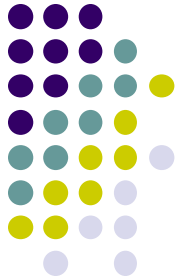
Example Usage:
Setting View Volume/Projection Type



```
void display()
{       // clear screen
        glClear(GL_COLOR_BUFFER_BIT);
        ……..
        // Set up camera position
        LookAt(0,0,1,0,0,0,0,1,0);
                    eye     at      up
        ……..
        // set up perspective transformation
        Perspective(fovy, aspect, near, far);

         ……..
        // draw something
        display_all();    // your display routine
}
```

Implementation



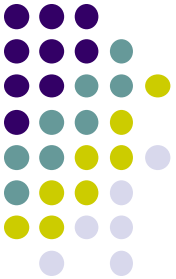
- Set modelview and projection matrices in application program
- Pass matrices to shader

```
void display( ){
    .....
    model_view = LookAt(eye, at, up);
    projection = Ortho(left, right, bottom,top, near, far);

    // pass model_view and projection matrices to shader
    glUniformMatrix4fv(matrix_loc, 1, GL_TRUE, model_view);
    glUniformMatrix4fv(projection_loc, 1, GL_TRUE, projection);
  .....
}
```

Build 4x4 projection matrix

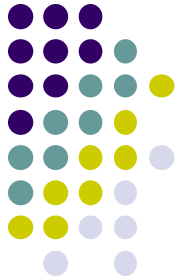
Implementation



- And the corresponding shader

```
in vec4 vPosition;
in vec4 vColor;
Out vec4 color;
uniform mat4 model_view;
Uniform mat4 projection;

void main( )
{
    gl_Position = projection * model_view * vPosition;
    color = vColor;
}
```



References

- Interactive Computer Graphics (6th edition), Angel and Shreiner
- Computer Graphics using OpenGL (3rd edition), Hill and Kelley