Classical and Computer Viewing

Adapted From:
Ed Angel
Professor of Emeritus of Computer Science
University of New Mexico
Planar Geometric Projections

Standard projections project onto a plane. **Projectors** are lines that either
- converge at a center of projection
- are parallel

Such projections preserve lines, but not necessarily angles.

*Nonplanar projections are needed for applications such as map construction.*
Classical Projections

- Front elevation
- Elevation oblique
- Plan oblique
- Isometric
- One-point perspective
- Three-point perspective
Taxonomy of Planar Projections

planar geometric projections

parallel

multiview
orthographic

axonometric
oblique

isometric
dimetric
trimetric

perspective

1 point
2 point
3 point
Parallel Projection

Object

Projector

Projection plane

DOP
Orthographic Projection

Projectors are orthogonal to projection surface
Multiview Orthographic Projection

Projection plane parallel to principal face
Usually form front, top, side views (6 possible)

isometric (not multiview orthographic view)
Axonometric Projections

Object rotated along one or more principal axes

classify by how many angles of a corner of a projected cube are the same

none: trimetric
two: dimetric
three: isometric

Projection plane
Types of Axonometric Projections

- Dimetric
- Trimetric
- Isometric
Oblique Projection

Arbitrary relationship between projectors and projection plane
Examples
Perspective Projection

Projectors coverge at center of projection
Vanishing Points

Parallel lines (not parallel to the projection plane) on the object converge at a single point in the projection (the *vanishing point*).

Drawing simple perspectives by hand uses these vanishing point(s).
Three-Point Perspective

No principal direction parallel to projection plane
Three vanishing points for cube
Two-Point Perspective

On principal direction parallel to projection plane
Two vanishing points for cube
One-Point Perspective

One principal face parallel to projection plane
One vanishing point for cube
Computer Viewing

There are three aspects of the viewing process, all of which are implemented in the pipeline,

- Positioning the camera
  - Setting the model-view matrix

- Selecting a lens
  - Setting the projection matrix (ortho vs. persp)

- Clipping
  - Setting the view volume
The Default OpenGL Camera (if you don’t specify one)

In OpenGL, initially the object and camera frames are the same

Default model-view matrix is an identity

The camera is located at origin and points in the negative z direction

Default projection type is Orthographic

OpenGL also specifies a *default view volume that is a cube with sides of length 2 centered at the origin*

Default projection matrix is an identity
Default Projection

Default projection is orthogonal
The LookAt Function

The GLU library contained the function gluLookAt (deprecated) to form the required modelview matrix through a simple interface

Replaced by LookAt() in `mat.h`

```c
mat4 mv = LookAt(vec4 eye, vec4 at, vec4 up);
```

3D position

3D Vector
LookAt

LookAt(eye, at, up)
OpenGL Orthogonal Viewing

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

near and far measured from camera
OpenGL Perspective: Frustum

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

near and far are distances from cam and should always be positive!!!
Using Field of View

With **Frustum** it is often difficult to get the desired view perspective (**fovy**, **aspect**, **near**, **far**) often provides a better interface

\[
\text{aspect} = \frac{w}{h}
\]

front plane
Using the Model View Matrix

In 3D graphics, the model-view matrix is used to
- Position the camera
- Build the scene

The projection matrix is used to define the view volume and to select a camera lens

Although these matrices are no longer part of the OpenGL state, you will create them in your application and pass them to shaders to be applied