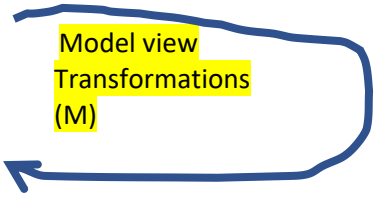
9/03/20

The CGI pipeline

Phase 1 – Model
Model-view volume
World coordinates, object space

Define Objects – using primitives ➔ library of objects (sphere)
Manipulate objects – duplicate; transform → translation, scaling, rotation



1) e.g., how to get an ellipse from a circle?
2) Egg from a sphere?

Uniform (in (x, y, z)) scale on circle/sphere ➔ sphere / circle
Non uniform ➔ 1, 2

Phase 2

Camera volume, eye volume camera view volume
Define the camera

1) Parallel projection camera
2) Perspective projection - aperture

Phase 3
NDC - Normalized Device Coordinates - standard form (OGL concern)

Phase 4
Port, in a window, on the screen = defined by the user the transformation is OGL concern

Step 1 – define objects – create a library

Use OpenGl primitives

Other primitives (outdated, laser-graphics)

Points

Points and lines

Raster – digital images

OGL Primitives

Geometry primitives

Vertices – points in 3-d

Lines- connecting vertices

Polygons

Only a few of these exist in OGL 3 and above

Raster

```
glBegin(Primitive_type);
        define vertices using glVertex
glEnd();
```

```
glVertex{2,3},{f, d, i, v}(x, y, [z]);
```

```
glBegin(GL_POLYGON);
        glVertex2f(-0.5, -0.5);
        glVertex2f(-0.5, 0.5);
        glVertex2f(0.5, 0.5);
        glVertex2f(0.5, -0.5);
glEnd();
```

```
glBegin(GL_POLYGON);
        glVertex{2,3},{f, d, i, v}(x, y, [z]);
glEnd();
```

Phase 2

Camera volume, eye volume camera view volume
Define the camera

1) Parallel projection camera
2) Perspective projection

One of the first things to do is to define the camera (projection type parameters)

By default the camera (lens, eye) is placed in the origin (0, 0, 0, 1)
Facing the -Z direction

Perspective
CG02-11 Assume an object (point) in (x, y, z, 1) all the projectors are going from a point through the lens to the projection plan (PP) Hit PP at a point (Xp, Yp, Zp, 1)
Eye Retina (PP), Digital camera sensor (e.g., CCD), Film in fil camera (PP is behind the lens)
Slide 12 – Perspective projection
Lens Center of projection (COP) all projectors pass through the COP
The PP (image plan) is in front of COP (e.g., a monitor)

CG03 – 9
The camera cannot see objects that are too close
Any object that is closer to the camera than the front
Clipping plane is invisible
Similarly the back-clipping plan – The view plan can be anywhere
CG 05-9 read

02/3/2020 (X,Z), (Y,Z) projections
LRBTNF

In Z we have the Near and Far point closer than near further than far invisible
In the X we cannot see left of Left or right of Right
In Y we cannot see above T or below B

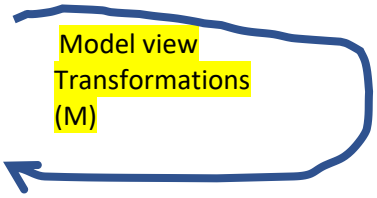
Defining LRBTNF for a real lens ? what type of volume is defined
Cut cone
Our model is a clipped (cut) pyramid – frustum.
For perspective projection the view volume is a frustum.
You will define the frustum (the view volume) only vertices that are inside
will be rendered.

In parallel projection all the projectors are parallel to the DOP vector.
Vertices are visible if hey inside the volume defined by LRBTNF
View volume is a box.
The default is parallel projection with (L, R, B, T, N, F) = (-1, 1, -1, 1, -1, 1)
Cube centered at the origin with edges of "size" 2.
Aerial photography

CG05 slide 9 – the default view volume is a cube due to parallel projection.

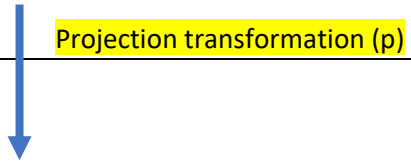
Phase 1 – Model
Model-view volume
World coordinates, object space

Define Objects – using primitives ➔ library of objects (sphere)
Manipulate objects – duplicate; transform → translation, scaling, rotation



3) e.g., how to get an ellipse from a circle?
4) Egg from a sphere?

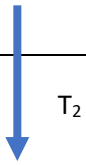
Uniform (in (x, y, z)) scale on circle/sphere ➔ sphere / circle
Non uniform ➔ 1, 2



Phase 2

Camera volume, eye volume camera view volume
Define the camera
3) Parallel projection camera
4) Perspective projection - aperture

Phase 3
NDC - Normalized Device Coordinates - standard form (OGL concern)



Phase 4
Port, in a window, on the screen = defined by the user the transformation is OGL concern

Step 1 – define objects – create a library
Use OpenGl primitives

Other primitives (outdated, laser-graphics)

- Points
- Points and lines
- Raster – digital images

OGL Primitives

- Geometry primitives
  - Vertices – points in 3-d
  - Lines- connecting vertices
  - Polygons
    - Only a few of these exist in OGL 3 and above
- Raster

```
glBegin(Primitive_type);
        define vertices using glVertex
glEnd();

glVertex{2,3},{f, d, i, v}(x, y, [z]);

glBegin(GL_POLYGON);
        glVertex2f(-0.5, -0.5);
        glVertex2f(-0.5, 0.5);
        glVertex2f(0.5, 0.5);
        glVertex2f(0.5, -0.5);
glEnd();


glBegin(GL_POLYGON);
        glVertex{2,3},{f, d, i, v}(x, y, [z]);
glEnd();
```

Phase 2

Camera volume, eye volume camera view volume

Define the camera

3) Parallel projection camera
4) Perspective projection

One of the first things to do is to define the camera (projection type parameters)

By default the camera (lens, eye) is placed in the origin (0, 0, 0, 1)
Facing the -Z direction

Perspective

CG02-11 Assume an object (point) in (x, y, z, 1) all the projectors are going from a point through the lens to the projection plan (PP) Hit PP at a point (Xp, Yp, Zp, 1)

Eye Retina (PP), Digital camera sensor (e.g., CCD), Film in fil camera (PP is behind the lens)

Slide 12 – Perspective projection

Lens Center of projection (COP) all projectors pass through the COP

The PP (image plan) is in front of COP (e.g., a monitor)

CG03 – 9

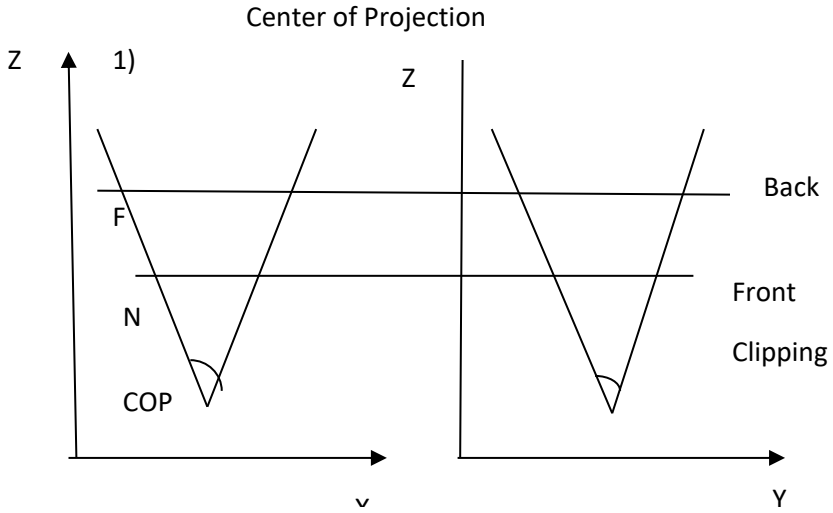
The camera cannot see objects that are too close

Any object that is closer to the camera than the front

Clipping plane is invisible

Similarly the back-clipping plan – The view plan can be anywhere

CG 05-9 read

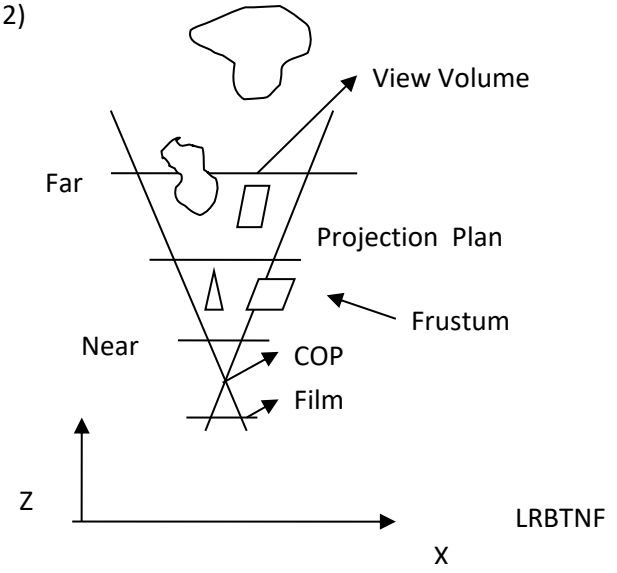


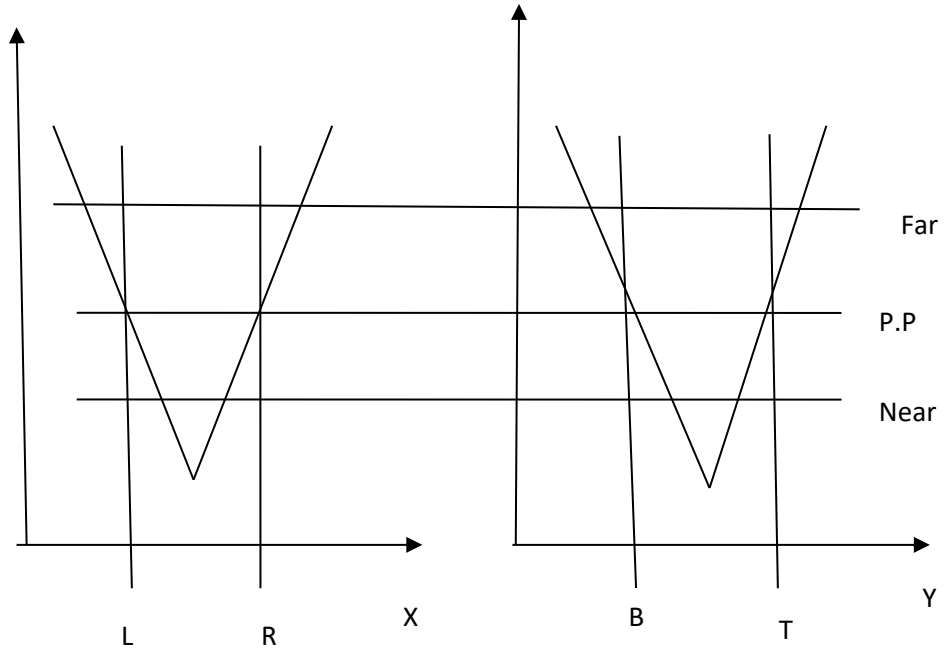
02/3/2020 (X,Z), (Y,Z) projections
LRBTNF

In Z we have the Near and Far point closer than near further than far invisible
In the X we cannot see left of Left or right of Right
In Y we cannot see above T or below B

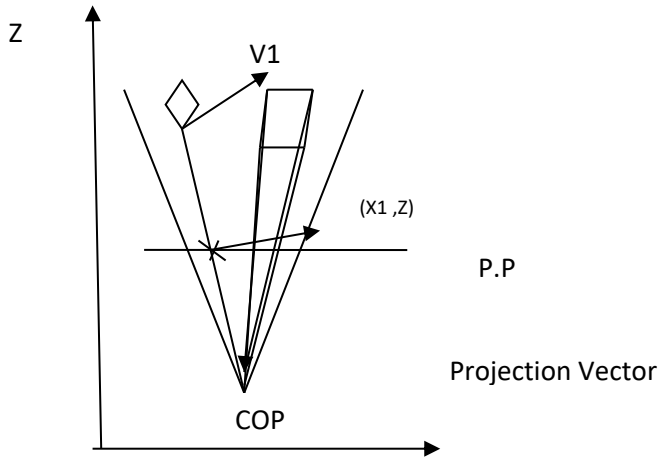
Defining LRBTNF for a real lens ? what type of volume is defined
Cut cone
Our model is a clipped (cut) pyramid – frustum.
For perspective projection the view volume is a frustum.
You will define the frustum (the view volume) only vertices that are inside
will be rendered.

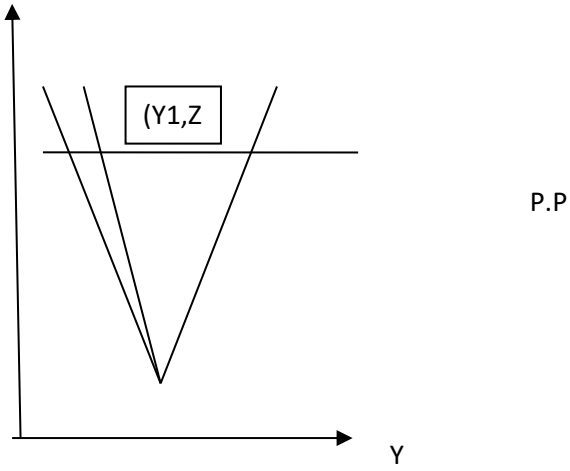
2)


View Volume
Far
Projection Plan
Frustum
Near
COP
Film
Z
X
LRBTNF


Far
P.P
Near
X
Y
L
R
B
T

3

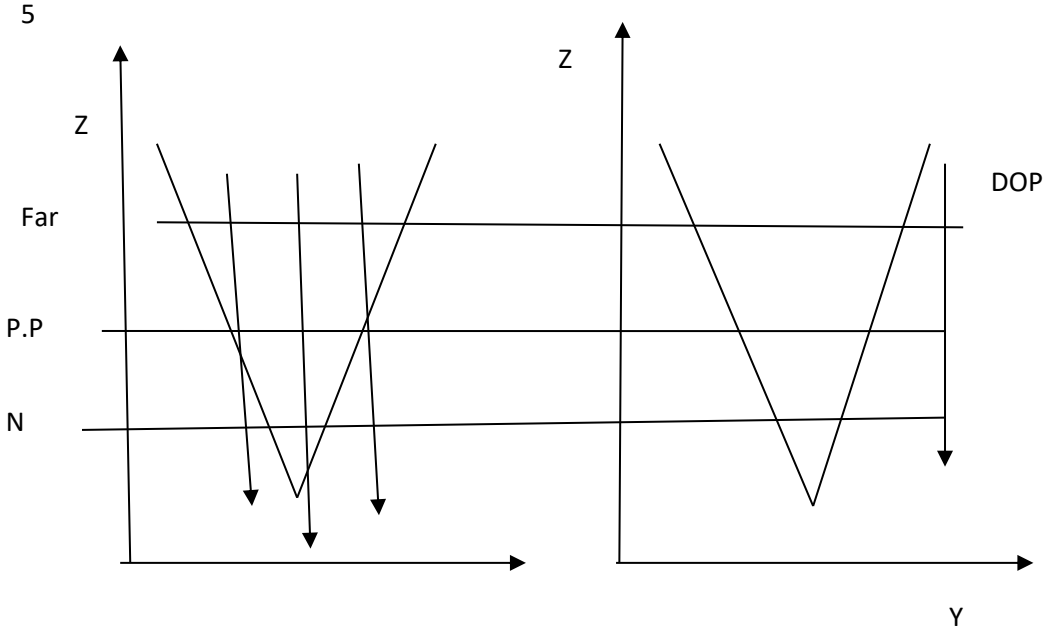




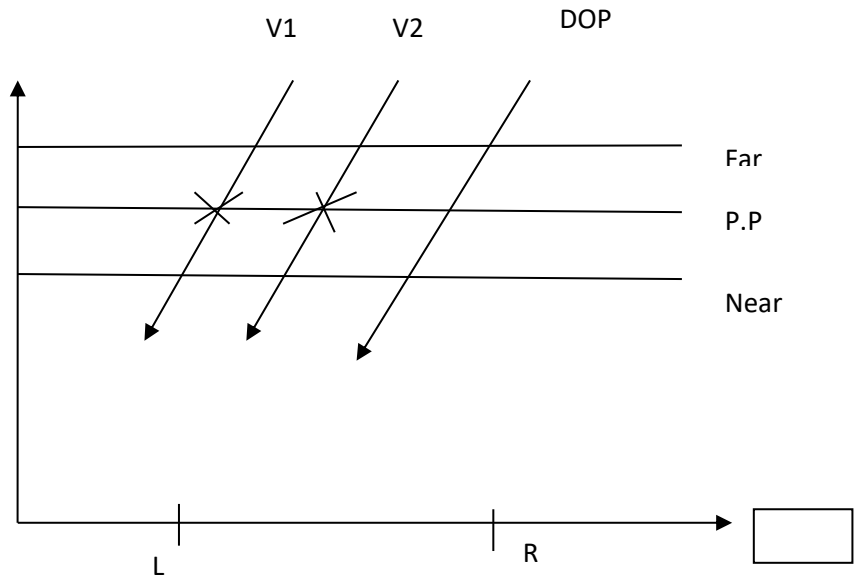
In parallel projection all the projectors are parallel to the DOP vector.
Vertices are visible if hey inside the volume defined by LRBTNF
View volume is a box.
The default is parallel projection with (L, R, B, T, N, F) = (-1, 1, -1, 1, -1, 1)
Cube centered at the origin with edges of “size” 2.
Aerial photography
CG05 slide 9 – the default view volume is a cube due to parallel projection.

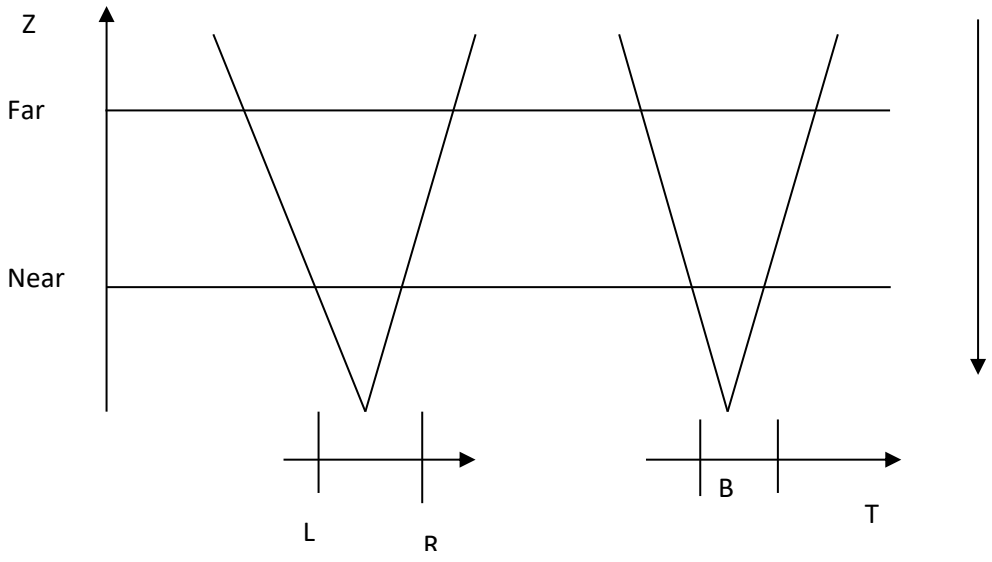
Parallel Projection

5



Parallel Projection(default)





LRBTNF

9/08/20

Goal is to be able to write rudimentary OGL program for object definition (Library of objects)

Start with defaults.
The default camera is parallel projection with (L, R, B, T, N, F) = (-1, 1, -1, 1, -1, 1)
Cube centered at the origin with edges of "size" 2.
Default port is the entire Window

Define objects. Using glBegin(); glVertex(), glEnd();
Understand the rendering using glFlush();

The software architecture.

Three libraries.
The C-GL – basic library (a class in many other environments e.g., Qt)
Primitives e.g., glBegin(); glVertex(), glEnd(); glFlush();

GLU – GL Utilities
Advanced primitives based on GL; shortcut; might be macros or functions using GL
Example, glFrustum() is used to define a perspective projection camera
gluPerspective is used to define a perspective projection camera

GLUT GL Utility Kit – Rudimentary Graphical User interface to OpenGL (to GL/GLU)
Define move resize windows, menus, mouse, KBD input

Alternatively using Qt as the GUI to OGL

Side note 2 – there is a New version of GLUT (include OGL and GLU) which can be executed under CodeBlocks. GL 2.x support?

Side note – 1
Graphics pipeline on workstations
PHIGS,
GL
OpenGl – up to 2.x
GPU
OpenGl 3, and above

GPU/CPU with emphasis on architecture
Vulkan and metal

GL/GLUT GUI

Event based programming (mouse click, KBD click, window resizing);
The OS logs numerous events
GLUT has access to the OS Event Queue – for some events it is generates call back functions
Example moving a window is an event, logged, and can (will) activate a related
Call back function. It is the programmer task to provide this function.
On several windows event glutDisplayFunc(); is invoked and executes the user supplied
Function (glutDisplayFunc(mydisplay); mydisplay

Modus Operands
Main()
Transfer line parameters to OGL
Have an Init function
Define Callback Functions
Go into an infinite loop waiting for events (glutMainLoop();

glutDisplayFunc(mydisplay); mydisplay is a "required" callback and you write it.

CG04 slide 18 is an example of generating a static polygon (square quad)
glFlush() is used for rendering