**clear information from last draw:**

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

**set background color:** glClearColor(R,G,B,1);

**switch to the drawing perspective:** glMatrixMode(GL\_MODELVIEW);

**reset the drawing perspective:** glLoadIdentity();

**set ambient color/light (natural color of a scene):**

GLfloat ambientColor[] = {R,G,B,1};

glLightModelfv(GL\_LIGHT\_MODEL\_AMBIENT, ambientColor);

**set light source of any structure:**

GLfloat lightColor0[] = {posL1, posL2, posL3, 1.0f};

GLfloat lightPos0[] = {0.0f, 8.0f, -4.0f, 1.0f};

glLightfv(GL\_LIGHT0,GL\_DIFFUSE, lightColor0);

glLightfv(GL\_LIGHT0, GL\_POSITION, lightPos0);

Here, glLightfv set light source properties.

GL\_LIGHT0 specifies the number of light, (at least eight lights are supported)

GL\_DIFFUSE specifies the type of light source

lightColor0 sets the color of the diffuse light(reflected light from object)

GL\_POSITION specifies position of light source

lightPos0 sets the light source position

**Binding Texture to a Texture Target:**

glBindTexture(GL\_TEXTURE\_2D, \_textureName);

GL\_TEXTURE\_2D is a texture target.

Textures have to be bound to apply operations to them. Since images are 2D arrays of pixels, it will be bound to the GL\_TEXTURE\_2D target.

**Filtering or Smoothing Image:**

glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_MIN\_FILTER, GL\_LINEAR);

glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_MAG\_FILTER, GL\_LINEAR);

Since texture coordinates are resolution independent, they won't always match a pixel exactly. This happens when a texture image is stretched beyond its original size or when it's sized down. OpenGL offers glTexParameteri() for that.

Here,

GL\_LINEAR: Returns the weighted average of the 4 pixels surrounding the given coordinates.

GL\_TEXTURE\_MIN\_FILTER: scaling image down.

GL\_TEXTURE\_MAG\_FILTER: scaling image up.

\*day/night image

**Sphere Drawing OpenGL function:**

GLUquadric \*quadric;

quadric = gluNewQuadric();

gluQuadricTexture(quadric, 50);

gluSphere(quadric,100,80,80);

Here,

The **gluQuadricTexture** function specifies whether quadrics are to be textured.

The **gluSphere** function draws a sphere of the given radius centered around the origin.

\*day/night image

**Fog Effect:**

GLfloat fogcolor[4]= {1.0f, 1.0f, 1.0f, 5};

glFogfv(GL\_FOG\_COLOR,fogcolor); **- sets fog color**

glFogf(GL\_FOG\_DENSITY,0.2); **-** **sets fog density (nonnegative and default 1)**

glFogi(GL\_FOG\_MODE,GL\_LINEAR);

glFogf(GL\_FOG\_START,2.0);

glFogf(GL\_FOG\_END,50.0);

glEnable(GL\_FOG);

**The linear function starts at 0 (here starts at 2) until we reach a certain distance (at 50), then linearly climbs to 1 at a second distance, then stays at 1.**

**Tree Leaf and Flag Image Update:**

GLuint leaf\_tex[10]; // image array

int leaf\_tex\_cnt=4, current\_leaf\_tex=0, leaf\_tex\_change\_time=250;

GLuint flag\_tex[10]; // image array

int flag\_tex\_cnt=9, current\_flag\_tex=0, flag\_tex\_change\_time=250;

void updateFlag()

void updateTreeLeaf()

These functions are invoked after every 250 ms.

**Key Functions**

|  |  |
| --- | --- |
| **Action** | **Key Press** |
| **Reset to Initial view** | **a** |
| **Light On/Off** | **l** |
| **Fog** | **f** |
| **Day/Night** | **n** |
| **Bottom views (multiple views)** | **b** |
| **Corner views (multiple views)** | **c** |
| **Left Rotation** | **q** |
| **Right Rotation** | **r** |
| **Upward Rotation** | **u** |
| **Downward Rotation** | **d** |
| **Moving upward** | **p** |
| **Moving downward** | **o** |
| **Top view** | **t** |
| **Moving left/right/front/rear** | **Arrow keys** |