OpenGL® Lectures
Glut Object Case Study

By
Tom Duff
Pixar Animation Studios
Emeryville, California
and
George Ledin Jr
Sonoma State University
Rohnert Park, California

Settings used for this case study

- When rendering the objects, depth-buffer is turned on to show the front view only.
  - glEnable(GL_DEPTH_TEST);
- When rendering solid objects, one light is enabled so that the shape of objects is highlighted.
  - The code to turn the light on:

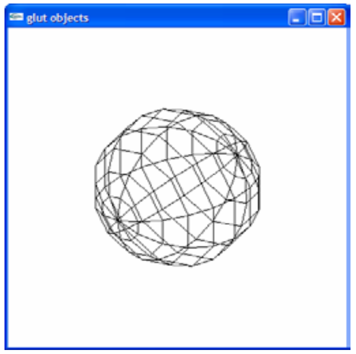
```
glEnable(GL_LIGHTING);          // enable the light
glEnable(GL_LIGHT0);            // turn on one light, light0
```

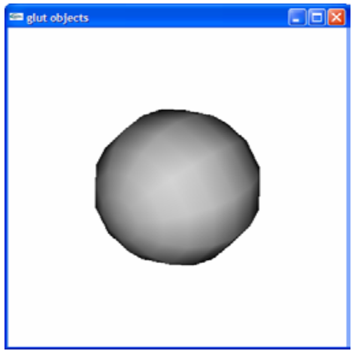
  Since we did not specify the light parameters, the OpenGL default parameters are used. In OpenGL default setting, the light0 is shining in –z direction.

- Different gluLookAt are used to adjust the camera to view the object.
- Different glOrtho values are used to adjust the size of object with respect to viewport.

Glut Objects - Sphere

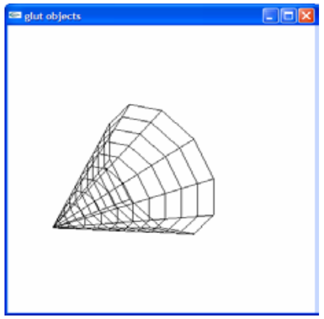
- void glutSolidSphere(GLdouble radius, GLint slices, GLint stacks);
- void glutWireSphere(GLdouble radius, GLint slices, GLint stacks);
  - Radius
    - The radius of the sphere.
  - slices
    - The number of subdivisions around the Z axis (similar to lines of longitude).
  - stacks
    - The number of subdivisions along the Z axis (similar to lines of latitude).
- **Example**
  - **glutSolidSphere(2,10,10);**
  - **glutWireSphere(2,10,10);**
  - **View: gluLookAt(1,1,1, 0,0,0, 0,1,0);**

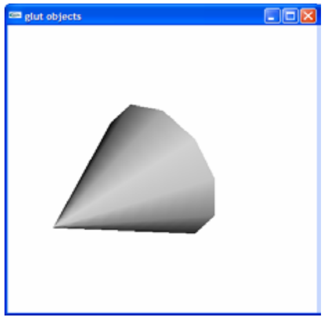




Glut Objects - Cone

- void glutSolidCone(GLdouble base, GLdouble height, GLint slices, GLint stacks);
- void glutWireCone(GLdouble base, GLdouble height, GLint slices, GLint stacks);
  - base
    - The radius of the base of the cone.
  - height
    - The height of the cone.
  - slices
    - The number of subdivisions around the Z axis.
  - stacks
    - The number of subdivisions along the Z axis.
- **Example**
  - **glutSolidCone(2,4,10,10);**
  - **glutWireCone(2,4,10,10);**
  - **View: gluLookAt(1,1,1, 0,0,0, 0,1,0);**





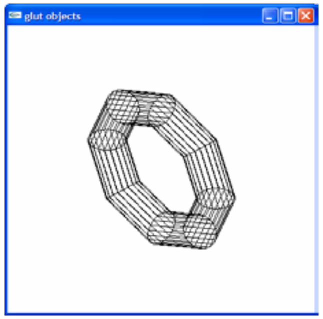
Glut Objects - Torus

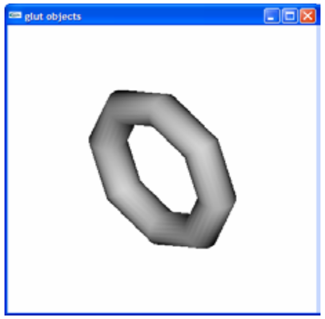
- void glutSolidTorus(GLdouble innerRadius, GLdouble outerRadius, GLint nsides, GLint rings);

- void glutWireTorus(GLdouble innerRadius, GLdouble outerRadius, GLint nsides, GLint rings);

  - innerRadius
    - Inner radius of the torus.
  - outerRadius
    - Outer radius of the torus.
  - nsides
    - Number of sides for each radial section.
  - rings
    - Number of radial divisions for the torus.

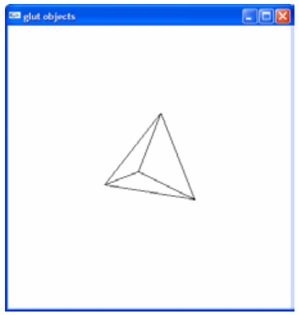
- **Example**
  - **glutSolidTorus(0.5,2,20,8);**
  - **glutWireTorus(0.5,2,20,8);**
  - **View: gluLookAt(1,1,1, 0,0,0, 0,1,0);**

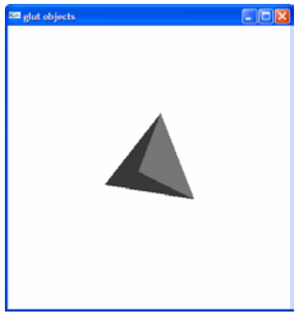




Glut Objects - Tetrahedron

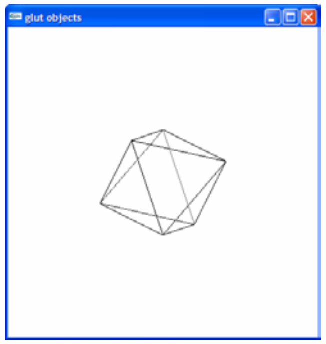
- void glutWireTetrahedron(void);
- void glutSolidTetrahedron(void);
  - Regular Polyhedral objects are defined with their vertices on a sphere of radius one
- View: gluLookAt(1,1,0.5,
  0,0,0,
  0,1,0);

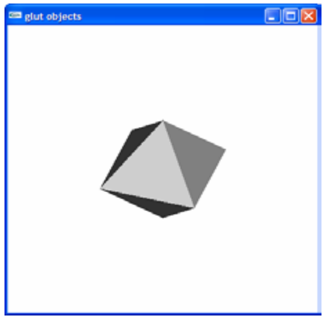




Glut Objects - Octahedron

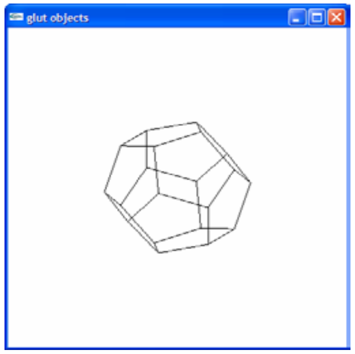
- void glutWireOctahedron(void);
- void glutSolidOctahedron(void);
  - Regular Polyhedral objects are defined with their vertices on a sphere of radius one
- gluLookAt(1,1,0.5,
  0,0,0,
  0,1,0);

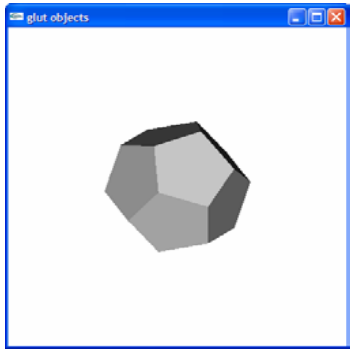




Glut Objects - Dodecahedron

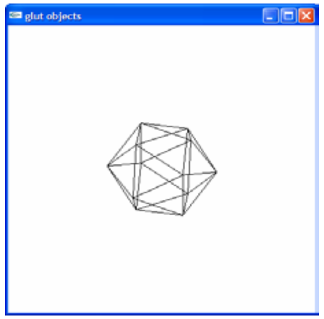
- void glutSolidDodecahedron(void);
- void glutWireDodecahedron(void);
  - Regular Polyhedral objects are defined with their vertices on a sphere of radius one
- gluLookAt(1,1,0.5,
  0,0,0,
  0,1,0);

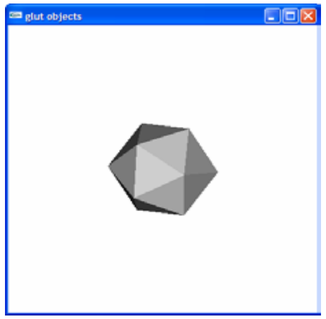




Glut Objects - Icosahedron

- void glutSolidIcosahedron(void);
- void glutWireIcosahedron(void);
  - Regular Polyhedral objects are defined with their vertices on a sphere of radius one
- gluLookAt(1,1,0.5,
  0,0,0,
  0,1,0);

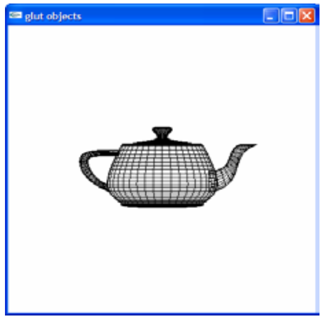


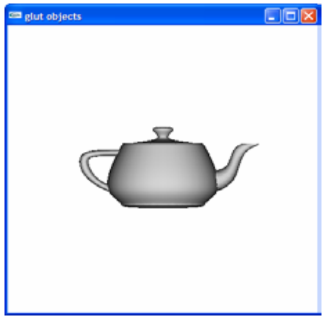


Glut Objects – Utah teapot

- void glutSolidUtahTeapot(size);
- void glutWireUtahTeapot(size);
  - The teapot has been used for many years for testing rendering algorithms.
- gluLookAt(0,0,0,

```
          0,0,-1,
          0,1,0);
```


glut objects


glut objects