gluSphere

NAME

gluSphere - draw a sphere

C SPECIFICATION

void **gluSphere**(GLUquadricObj **qobj*, GLdouble *radius*, GLint *slices*, GLint *stacks*)

PARAMETERS

qobj Specifies the quadrics object (created with **gluNewQuadric**).

radius Specifies the radius of the sphere.

slices Specifies the number of subdivisions around the *z* axis (similar to lines of longitude).

stacks Specifies the number of subdivisions along the *z* axis (similar to lines of latitude).

DESCRIPTION

gluSphere draws a sphere of the given radius centered around the origin. The sphere is subdivided around the *z* axis into slices and along the *z* axis into stacks (similar to lines of longitude and latitude).
If the orientation is set to **GLU_OUTSIDE** (with **gluQuadricOrientation**), then any normals generated point away from the center of the sphere. Otherwise, they point toward the center of the sphere.
If texturing is turned on (with **gluQuadricTexture**), then texture coordinates are generated so that *t* ranges from 0.0 at *z = -radius* to 1.0 at *z = radius* (*t* increases linearly along longitudinal lines), and *s* ranges from 0.0 at the +*y* axis, to 0.25 at the +*x* axis, to 0.5 at the -*y* axis, to 0.75 at the -*x* axis, and back to 1.0 at the +*y* axis.

gluDisk

NAME

gluDisk - draw a disk

C SPECIFICATION

void **gluDisk**(GLUquadricObj **qobj*, GLdouble *innerRadius*, GLdouble *outerRadius*, GLint *slices*, GLint *loops*)

PARAMETERS

qobj Specifies the quadrics object (created with **gluNewQuadric**).

innerRadius Specifies the inner radius of the disk (may be 0).

outerRadius Specifies the outer radius of the disk.

slices Specifies the number of subdivisions around the *z* axis.

loops Specifies the number of concentric rings about the origin into which the disk is subdivided.

DESCRIPTION

gluDisk renders a disk on the *z* = 0 plane. The disk has a radius of *outerRadius*, and contains a concentric circular hole with a radius of *innerRadius*. If *innerRadius* is 0, then no hole is generated. The disk is subdivided around the *z* axis into slices (like pizza slices), and also about the *z* axis into rings (as specified by *slices* and *loops*, respectively).
With respect to orientation, the +*z* side of the disk is considered to be "outside" (see "gluQuadricOrientation"). This means that if the orientation is set to **GLU_OUTSIDE**, then any normals generated point along the +*z* axis. Otherwise, they point along the -*z* axis.
If texturing is turned on (with **gluQuadricTexture**), texture coordinates are generated linearly such that where *r = outerRadius* , the value at (*r*, 0, 0) is (1, 0.5), at (0, *r*, 0) it is (0.5, 1), at (-*r*, 0, 0) it is (0, 0.5), and at (0, -*r*, 0) it is (0.5, 0).

gluCylinder

NAME

gluCylinder - draw a cylinder

C SPECIFICATION

void **gluCylinder**(GLUquadricObj **qobj*, GLdouble *baseRadius*, GLdouble *topRadius*, GLdouble *height*, GLint *slices*, GLint *stacks*)

PARAMETERS

qobj	Specifies the quadrics object (created with **gluNewQuadric**).
baseRadius	Specifies the radius of the cylinder at $z = 0$.
topRadius	Specifies the radius of the cylinder at $z = height$.
height	Specifies the height of the cylinder.
slices	Specifies the number of subdivisions around the z axis.
stacks	Specifies the number of subdivisions along the z axis.

DESCRIPTION

gluCylinder draws a cylinder oriented along the z axis. The base of the cylinder is placed at $z = 0$, and the top at $z = height$. Like a sphere, a cylinder is subdivided around the z axis into slices, and along the z axis into stacks.
Note that if *topRadius* is set to zero, then this routine will generate a cone.
If the orientation is set to **GLU_OUTSIDE** (with **gluQuadricOrientation**), then any generated normals point away from the z axis. Otherwise, they point toward the z axis.
If texturing is turned on (with **gluQuadricTexture**), then texture coordinates are generated so that t ranges linearly from 0.0 at $z = 0$ to 1.0 at $z = height$, and s ranges from 0.0 at the $+y$ axis, to 0.25 at the $+x$ axis, to 0.5 at the $-y$ axis, to 0.75 at the $-x$ axis, and back to 1.0 at the $+y$ axis.

11.1 glutSolidSphere, glutWireSphere

`glutSolidSphere` and `glutWireSphere` render a solid or wireframe sphere respectively.

Usage

```
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).

Description

Renders a sphere centered at the modeling coordinates origin of the specified `radius`. The sphere is subdivided around the Z axis into slices and along the Z axis into stacks.

11.2 glutSolidCube, glutWireCube

`glutSolidCube` and `glutWireCube` render a solid or wireframe cube respectively.

Usage

```
void glutSolidCube(GLdouble size); void glutWireCube(GLdouble size);
```

Description

`glutSolidCube` and `glutWireCube` render a solid or wireframe cube respectively. The cube is centered at the modeling coordinates origin with sides of length `size`.

11.3 glutSolidCone, glutWireCone

`glutSolidCone` and `glutWireCone` render a solid or wireframe cone respectively.

Usage

```
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.

Description

`glutSolidCone` and `glutWireCone` render a solid or wireframe cone respectively oriented along the Z axis. The base of the cone is placed at Z = 0, and the top at Z = `height`. The cone is subdivided around the Z axis into slices, and along the Z axis into stacks.

11.4 glutSolidTorus, glutWireTorus

`glutSolidTorus` and `glutWireTorus` render a solid or wireframe torus (doughnut) respectively.

Usage

```
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.

Description

`glutSolidTorus` and `glutWireTorus` render a solid or wireframe torus (doughnut) respectively centered at the modeling coordinates origin whose axis is aligned with the Z axis.

11.9 glutSolidTeapot, glutWireTeapot

`glutSolidTeapot` and `glutWireTeapot` render a solid or wireframe teapot respectively.

Usage

```
void glutSolidTeapot(GLdouble size); void glutWireTeapot(GLdouble
size);
```

`size`
: Relative size of the teapot.

Description

`glutSolidTeapot` and `glutWireTeapot` render a solid or wireframe teapot respectively. Both surface normals and texture coordinates for the teapot are generated. The teapot is generated with OpenGL evaluators.