Quadrics et al

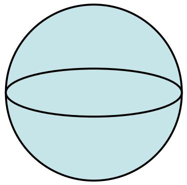
CS116B
Chris Pollett
Jan 31, 2005.

Outline

- Quadric Surfaces
- SuperQuadrics
- OpenGL

Quadrics Surfaces

- Quadrics are a frequently used class of surfaces.
- They include spheres, ellipsoids, torii, paraboloids, and hyperboloids.
- They get their name because they are described by second degree equations.



Spheres

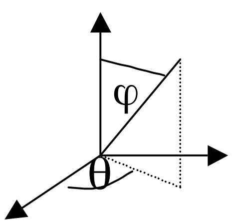
- The sphere of radius r can be described as those points satisfying the equation: $x^2+y^2+z^2=r^2$. (non-parametric equation)
- It can also be described as those points satisfying:

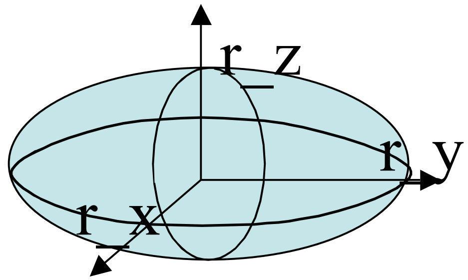
$x(\varphi,\theta) = r*\cos\varphi*\cos\theta$, $-\pi/2 <= \varphi <= \pi/2$

$y(\varphi,\theta) = r*\cos\varphi*\sin\theta$, $-\pi <= \theta <= \pi$

$z(\varphi,\theta) = r*\sin\theta$

I wrote x,y,z as above to emphasize we have a mapping from (φ,θ) to $(r*\cos\varphi*\cos\theta, r*\cos\varphi*\sin\theta, r*\sin\theta)$





Ellipsoid

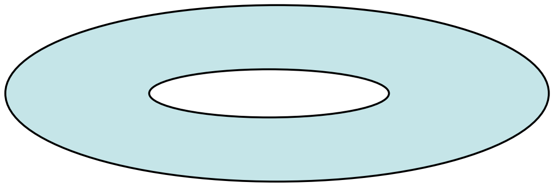
- Can view as a squashed sphere.
- More, formally, we have three radii: r_x, r_y, r_z and the ellipsoid is described by those points satisfying:

  $(x/r_x)^2 + (y/r_y)^2 + (z/r_z)^2 = 1.$
- A sphere is an ellipsoid where $r_x=r_y=r_z$.
- The parametric equations for an ellipsoid are:

  $x(\varphi,\theta) = r\_x*\cos\varphi*\cos\theta, \quad -\pi/2 <= \varphi <= \pi/2$

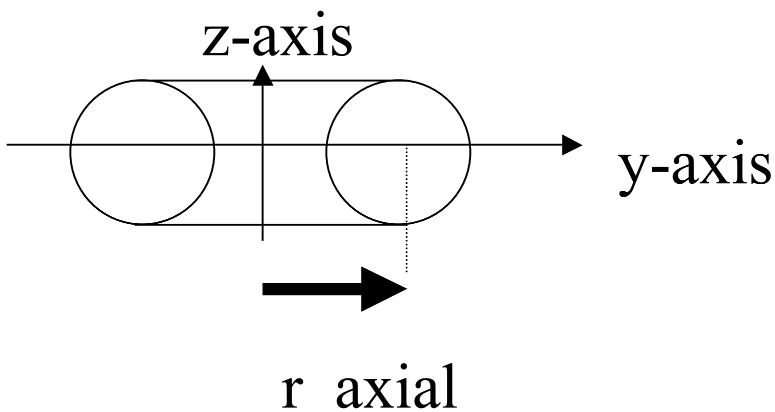
  $y(\varphi,\theta) = r\_y*\cos\varphi*\sin\theta, -\pi <= \theta <= \pi$

  $z(\varphi,\theta) = r\_z*\sin\theta$



Torus

- Basically, a fancy word for a doughnut
- Can make a torus by rotating a circle along a perpendicular circle of some fixed radius:



- One can do this with the equations:

  $(y-r_{axial})^2+z^2=r^2$ or $x=(r_{axial}+r*\cos\varphi)*\cos\theta$, $y=(r_{axial}+r*\cos\varphi)*\sin\theta$, $z=r*\sin\varphi$.

SuperQuadrics

- These are generalizations of quadrics.
- We add some additional parameters to the quadric equations which allows us to tweak the basic shapes.
- The equations will generally not be quadratic in these new parameters.

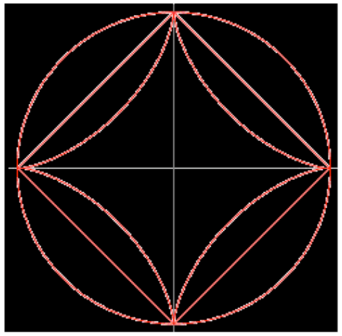
Superellipse

- A superellipse (a 2D object) is given by the equation:

  $(x/r_x)^{2/s} + (y/r_y)^{2/s} = 1$ or

  $x = r_x \cos^s \theta$, $y = r_y \sin^s \theta$

- Figure below shows different s>=1



Superellipsoid

- A superellipsoid is a generalization of a superellipse to 3D.
- Equations are:

$[(x/r_x)^{2/s\_2} + (y/r_y)^{2/s\_2}]^{s\_2/s\_2} + (z/r_z)^{2/s\_1} = 1$

or

$x = r_x * \cos^{s\_1} \varphi * \cos^{s\_2} \theta,$

$y = r_y * \cos^{s\_1} \varphi * \sin^{s\_2} \theta,$

$z = r_z * \sin^{s\_1} \varphi$

GLUT Quadrics

- We now discuss how to draw quadrics using GLUT.
- To begin, for spheres the functions are:

  glutWireSphere(r, nLongitudes, nLatitudes);

  or

  glutSolidSphere(r, nLongitudes, nLatitudes);
- Obviously, r is the radius (it is a double)
- nLongitudes is the number of circles through both poles to be used in the sphere.
- nLatitudes is the number of circles parallel to the equator to be used in the sphere.

More Glut Quadrics

- Cones can be drawn with:

  glutWireCone(rBase, height, nLongitudes, nLatitudes);

  or

  glutSolidCone(rBase, height, nLongitudes, nLatitudes);
- Torii can be drawn with:

  glutWireTorus(rCrossSection, rAxial, nConcentrics, nRadialSlices);

  or

  glutSolidTorus(rCrossSection, rAxial, nConcentrics, nRadialSlices);
- rCrossSection is the r of before, rAxial is r_axial.
- nConcentrics - num circles centered on z axis. nRadialSlices -circles used in the axial rotation

Famous GLUT Surface

- One of the first surfaces to be modeled (by hand) as a polygon mesh in a computer graphics system was a teapot (Martin Newell 1975).
- This teapot still exists and can be drawn using:

  glutWireTeapot(size); or glutSolidTeapot(size);

GLU Quadrics

- GLU provides slightly more flexible functions for drawing quadrics.
- Basic sequence of calls looks like:

```
GLUquadric *sphere1;
sphere1 = gluNewQuadric();
gluQuadricDrawStyle(sphere1, GLU_LINE); // wireframe
gluSphere(sphere1, r, nLongitudes, nLatitudes);
```

- Some draw styles are GLU_POINT (just points), GLU_SILHOUETTE (wireframe less shared edge for coplanar facets) and GLU_FILL (like solid)

More GLU Quadrics

- Some other quadrics available are:
  - gluCylinder(name, rBase, rTop, height, nLongitudes, nLatitudes);
  - gluDisk(name, rInner, rOuter, nRadii, nRings); //can use for flat disks with or without a hole
  - gluPartialDisk(name, rInner, rOuter, nRadii, nRings, startAngle, sweepAngle);

More GLU Quadric functions

- Once we are done with a quadric we can reclaim its memory with:

  gluDeleteQuadric(name);
- We can define the front and back direction for the facet's normal vectors with:

  gluQuadricOrientation(name, normalVectorDirection);

  //GLU_INSIDE

  //GLU_OUTSIDE

  gluQuadricNormals(name, generationMode);

  // GLU_NONE, GLU_FLAT, GLU_SMOOTH.
- Finally,we can define a callback that is called if an error occurs during the draw of the quadric with:

  gluQuadricCallback(name, GLU_ERROR, function);