Geometric Shapes: Blobby Shapes

Module 3
Lecture 9

Previously we learnt

$G: f(x,y,z) \geq 0$

$G_3 = G_1 \cup G_2 : f_3 = f_1 \vee f_2 = \max(f_1, f_2) \geq 0$ Union

$G_3 = G_1 \cap G_2 : f_3 = f_1 \wedge f_2 = \min(f_1, f_2) \geq 0$ Intersection

$G_3 = -G_1 : f_3 = -f_1$ Complement

$G_3 = G_1 \setminus G_2 : f_3 = f_1 \setminus f_2 = \min(f_1, -f_2) \geq 0$ Subtraction

Example:

$G_5 = G_1 \cup ((G_2 \cap G_3) \setminus G_4)$:

$f_5 = f_1 \vee ((f_2 \wedge f_3) \setminus f_4) = \max(f_1, \min(\min(f_2, f_3), -f_4)) \geq 0$

Blobby Shapes

- Blobby function

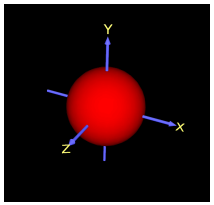
$g = f(x,y,z) = ae^{-r}$ $f(x,y,z) \geq 0$?

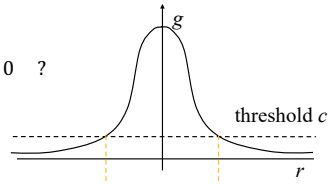
where

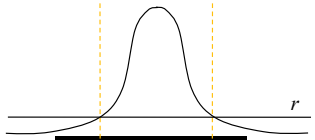
$r = (x - x_b)^2 + (y - y_b)^2 + (z - z_b)^2$

- Solid blob

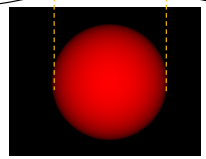
$g = f(x,y,z) = ae^{-r} - c \geq 0$







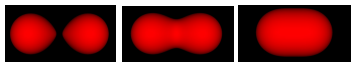
Animated threshold

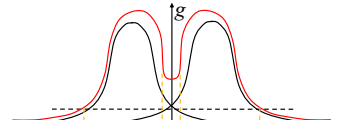


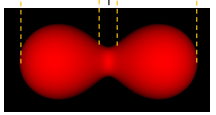
Blobby Shapes

Multiple blobs

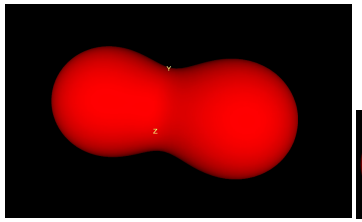
$g = f(x,y,z) = \sum_i a_i e^{-r_i} - c \geq 0$



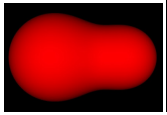


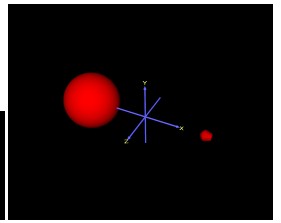


Moving centres of the blobs

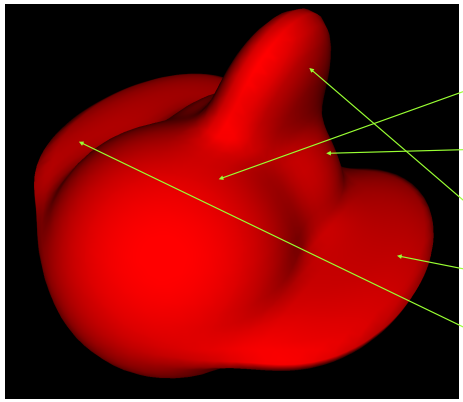


Different sizes of the blobs





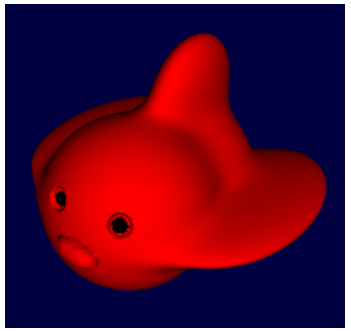
Blobby Shapes

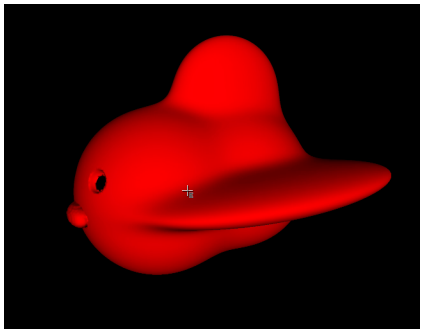


$$g = f(x,y,z) =$$
$$2e^{-\left((x+1)^2+y^2+z^2\right)}$$
$$+e^{-\left((x-1)^2+y^2+z^2\right)}$$
$$+0.3e^{-\left((x-1)^2+(y-2)^2+5z^2\right)}$$
$$+0.3e^{-\left((0.5(x-1))^2+(3y)^2+(z-2)^2\right)}$$
$$+0.3e^{-\left((0.5(x-1))^2+(3y)^2+(z+2)^2\right)}$$
$$-0.05 \geq 0$$

Blobby shapes

Applying min/max CSG operations to the blobby shape





Summary

- Blobby shapes are an example how a different representation can be incorporated into a family of surfaces and solid objects defined by implicit functions and inequalities (variants of implicit functions), respectively.
- In contrast to min/max Set-Theoretic Operations, blobby shapes allow for smooth merging objects together implemented as algebraic addition of individual blobby functions.

Geometric Shapes: How to draw curves, surfaces and solids.

Module 3
Lecture 10

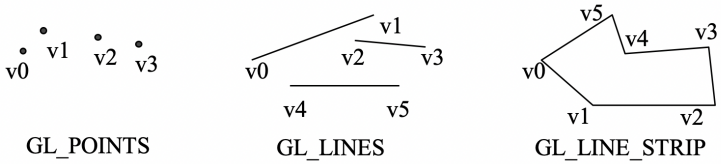
How to draw a curve?

- Linear interpolation
- Parametric representations are the most suitable for it

```
x1=x(0); y1=y(0); z1=z(0);
delta=1/sampling_resolution;
for (u=delta; u<=1; u=u+delta;)
{
      x2=x(u), y2=y(u), z2=z(u);
      draw_line (x1,y1,z1,x2,y2,z2);
      x1=x2; y1=y2; z1=z2;
}
```


Drawing points and lines in OpenGL

```
glBegin(GL_PrimitiveType);
     glVertex2f(0.0, 0.0);
     glVertex2f(0.0, 3.0);
     glVertex2f(3.0, 3.0);
     glVertex2f(3.0, 0.0);
glEnd();
```

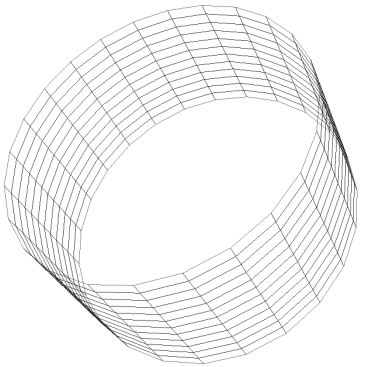


How to draw surfaces in polygon-based software?

- Wire-frame Representation
- Polygonization

Wire-frame of parametric surface

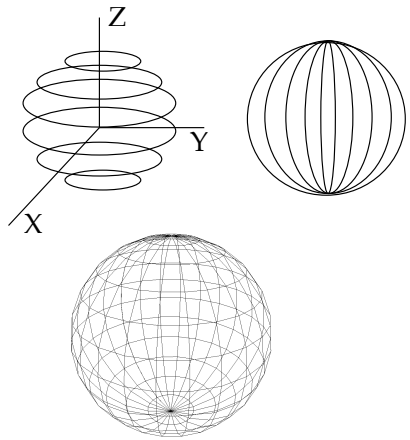
$$x = x(u, v)$$
$$y = y(u, v)$$
$$z = z(u, v)$$
$$u_1 \le u \le u_2$$
$$v_1 \le v \le v_2$$



Wire-frame of a sphere

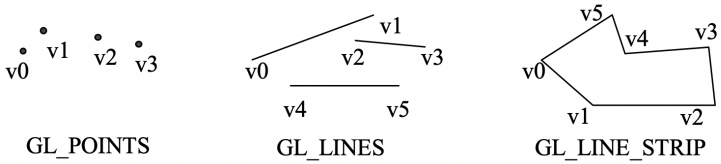
$$x = r\cos u\cos v$$
$$y = r\sin u\cos v$$
$$z = r\sin u$$
$$-\pi \le u \le \pi$$
$$-\pi/2 \le v \le \pi/2$$

Nested *for_loops* in u and v



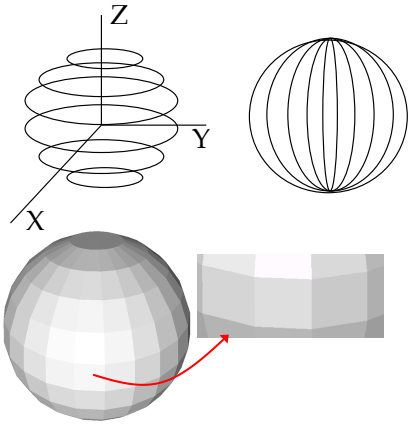
Drawing lines in OpenGL

```
glBegin(GL_PrimitiveType);
     glVertex2f(0.0, 0.0);
     glVertex2f(0.0, 3.0);
     glVertex2f(3.0, 3.0);
     glVertex2f(3.0, 0.0);
glEnd();
```



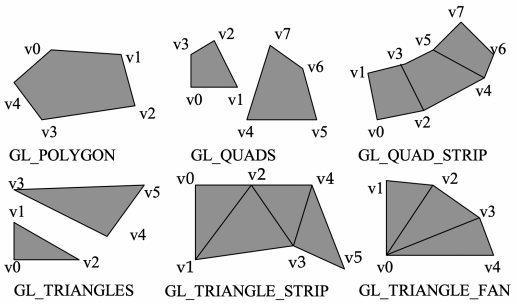
Polygonization of a sphere

$$x = r\cos u\cos v$$
$$y = r\sin u\cos v$$
$$z = r\sin v$$
$$-\pi \le u \le \pi$$
$$-\pi/2 \le v \le \pi/2$$



Displaying polygons in OpenGL

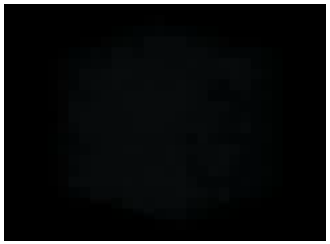
```
glBegin(GL_PrimitiveType);
     glVertex2f(0.0, 0.0);
     glVertex2f(0.0, 3.0);
     glVertex2f(3.0, 3.0);
     glVertex2f(3.0, 0.0);
glEnd();
```

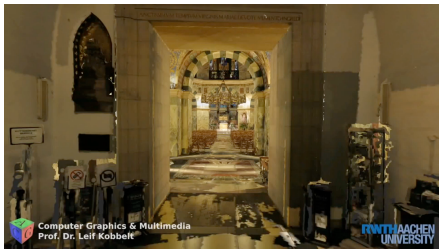


How to draw a solid shape?

- Voxel graphics
- Surface rendering

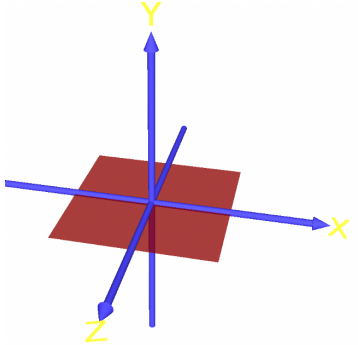
Voxel and point graphics



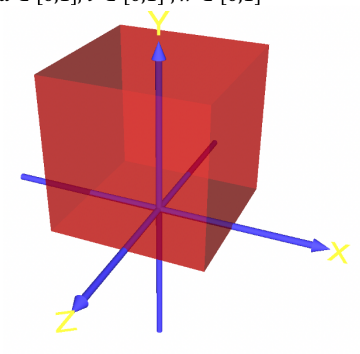


How to Draw Parametric Solids

$P = P(u, v)$
$u \in [0,1], v \in [0,1]$



$P = P(u, v, w)$
$u \in [0,1], v \in [0,1], w \in [0,1]$



How to Draw Parametric Solids

$P = P(u, v)$
$u \in [0,1], v \in [0,1]$

$P = P(u, v, w)$
$u \in [0,1], v \in [0,1], w \in [0,1]$

$P = P(u, v, w)$
$u \in [0,1], v \in [0,1], w_1 = 0$

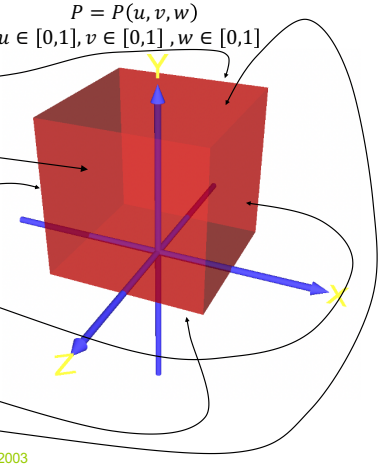
$P = P(u, v, w)$
$u \in [0,1], v \in [0,1], w_1 = 1$

$P = P(u, v, w)$
$u = 0, v \in [0,1], w \in [0,1]$

$P = P(u, v, w)$
$u = 1, v \in [0,1], w \in [0,1]$

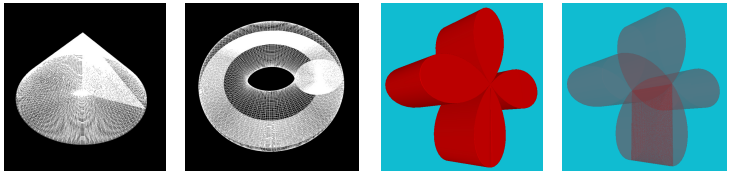
$P = P(u, v, w)$
$v \in [0,1], v = 0, w \in [0,1]$

$P = P(u, v, w)$
$v \in [0,1], v = 1, w \in [0,1]$



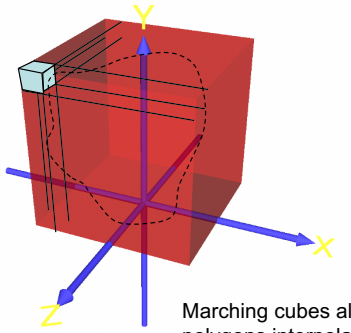
How to Draw Parametric Solids

- This method works only if the solid has up to 6 bounding surfaces, i.e., for each of the 3 parameters the domain is defined by min and max values
- Some redundant inner bounding surfaces and curves can be displayed. Though these artifacts cannot be seen from outside the solids unless their surface is defined transparent, they ideally have to be excluded from the final polygonization. This may be not a trivial task.



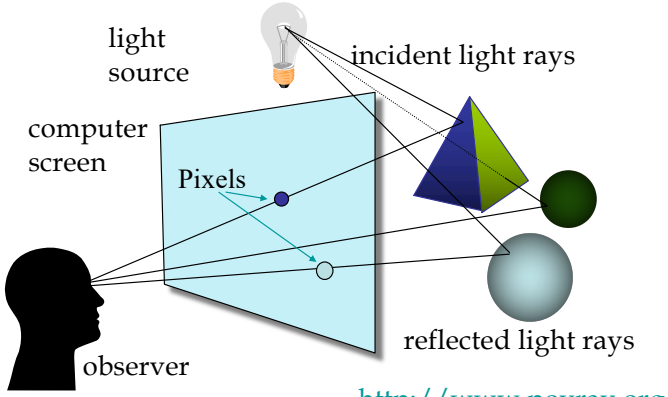
How to Draw Implicit Solids

$$f(u,v,w) \geq 0$$
$$x \in [x_1, x_2], y \in [y_1, y_2], z \in [z_1, z_2]$$



Marching cubes algorithm is extracting polygons interpolating the surface

Ray Tracing of Surfaces and Solids



http://www.povray.org

Intersection

$$\begin{cases} f(x,y,z) = 0 \\ x = x_1 + u(x_2 - x1) \\ y = y_1 + u(y_2 - y1) \\ x = z_1 + u(z - z1) \\ u \in [0, \quad big\ number] \end{cases}$$

Substitute x, y and z into the first equation and solve it in terms of u

Compute x, y, z

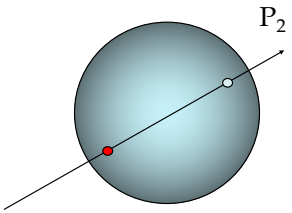
Intersection

- Intersection with a sphere

$$
\begin{cases}
r^2 - x^2 - y^2 - z^2 = 0 \\
x = x_1 + (x_2 - x_1)u \\
y = y_1 + (y_2 - y_1)u \\
z = z_1 + (z_2 - z_1)u
\end{cases}
$$

Quadric equation has 2 solutions u_1 and u_2

Select the smaller value of u since this point will be closer to P_1



Summary

- Line interpolation and polygonization dominate in visualization
- Point-based rendering and voxel (volume) graphics evolve rapidly
- Fast rendering can be achieved by optimal selection of the domain (parameters or bounding box) and sampling resolution