Lab 5: Morphing

Xavier Tan

Lab group: SSP5

Attendance Number : 18

Surface number: 18 and 23.

Software used : BSContact

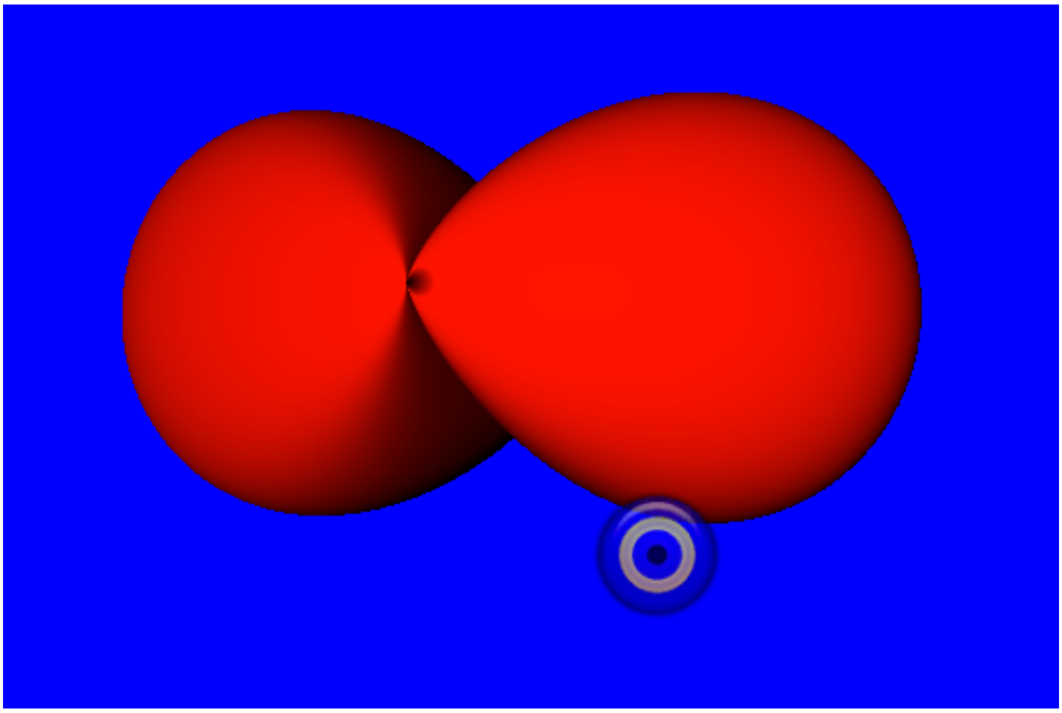
Filename: animation.wrl

Shape 1(18)

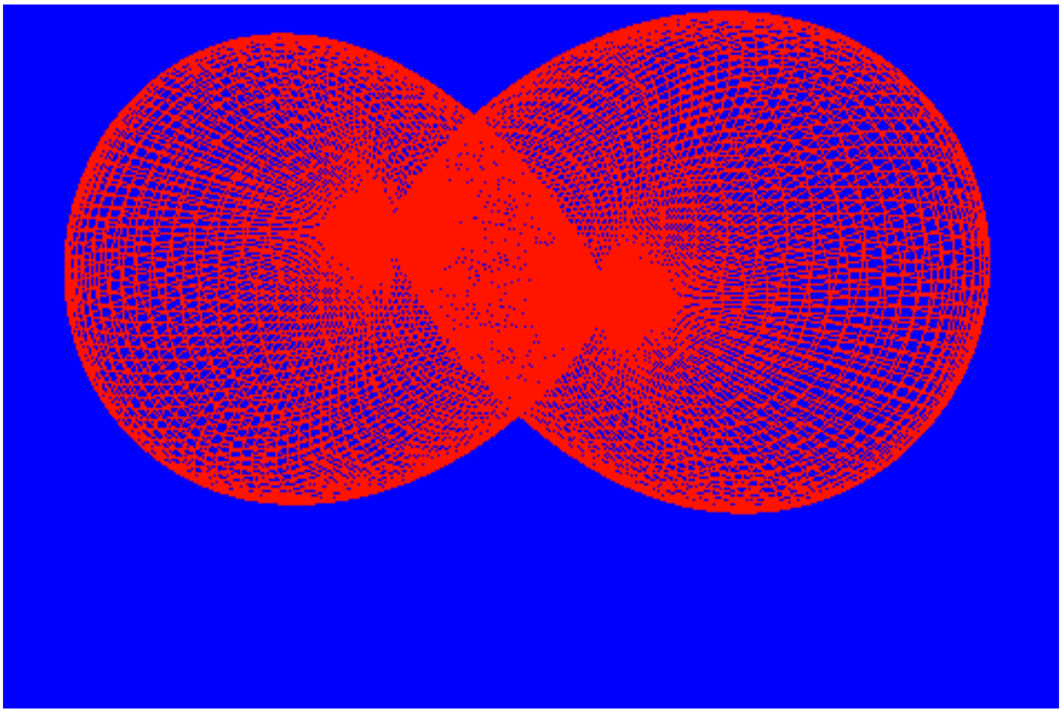
Given:
x = (cos(0.25q) +1)cos(j);
y = sin(0.25q)cos(j);
z = sin(j);
0<q<8*pi 0<j<2*pi

To re parameterise to q and j to u and v where u,v has domain [0 1]
Change q to u*8*pi, j to v*2*pi), resulting:

x = (cos(0.25*u*8*pi) +1)*cos(v*2*pi);
y=sin(0.25*u*8*pi)*cos(v*2*pi);
z=sin(2*pi*u)*sin(pi*v);



Shape 1, number 18



Shape 1, wireframe mode

Shape 2(23)

Given:

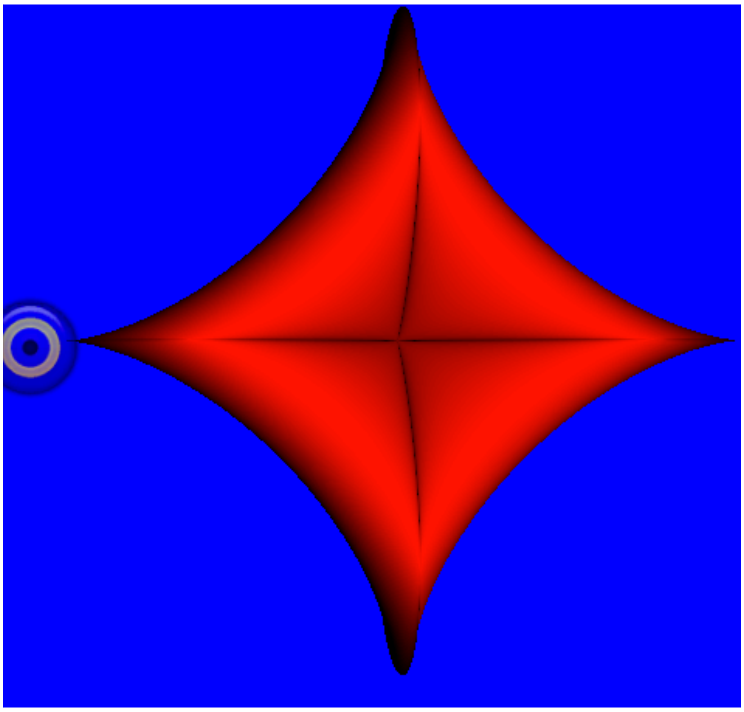
$x = 2(\cos 8q)^3(\sin j)^5$

$y = 2(\sin 8q)^3(\sin j)^5$

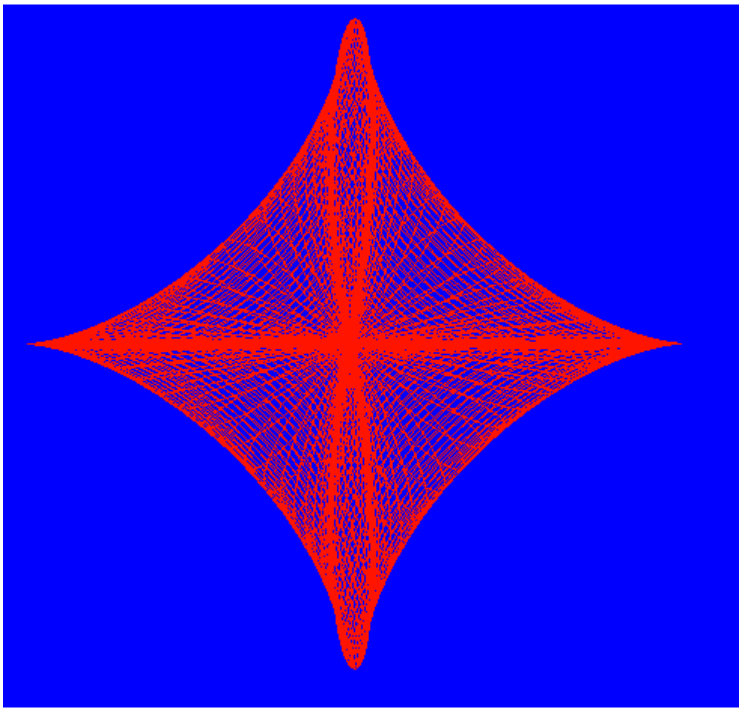
$z = 2(\sin j)^5 \cos(j)$

0<q<pi/4 0<j<pi

To re parameterise to q and j to u and v where u,v has domain [0 1]
Change q to u*¼ *pi, j to v*pi, resulting:
x = 2*(cos(8*pi*1/4*u)^3*sin(pi*v)^5);
y = 2*(sin(8*pi*1/4*u)^3*sin(pi*v)^5);
z = 2*(sin(pi*v)^5*cos(pi*v));

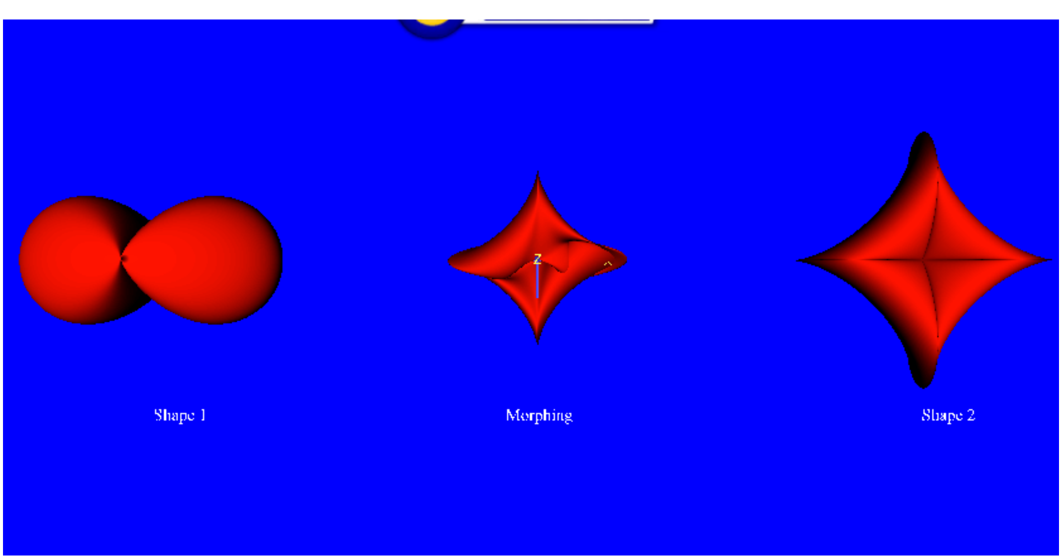


Shape 2, number 23

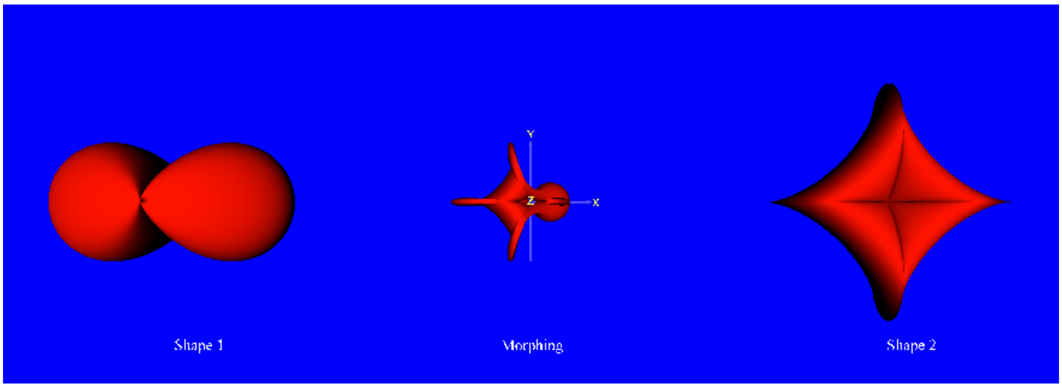


shape 2 wireframe mode

Morphing



All 3 Shapes, t is almost 1



Shape morphing to shape 2, t is approximate = 0.5

File Used :

Equation used:

```
function parametric_x(u,v,w,t)
{ x1=(cos(0.25*u*8*pi) +1)*cos(v*2*pi);
  x2=2*(cos(8*pi*1/4*u)^3*sin(pi*v)^5);
  return x1+(x2-x1)*t;       }

function parametric_y(u,v,w,t)
{ y1=sin(0.25*u*8*pi)*cos(v*2*pi);
  y2=2*(sin(8*pi*1/4*u)^3*sin(pi*v)^5);
  return y1+(y2-y1)*t;       }

function parametric_z(u,v,w,t)
{ z1=sin(v*2*pi);
  z2=2*(sin(pi*v)^5*cos(pi*v));  return z1+(z2-z1)*t;
 }"
```

Where x1,y1,z1 is the first shape and x2,y2,z2 is the second shape.

The transformation node is used to display the 2 original shapes where the one displaced 6 units to the negative x axis is the 1st shape, and the other displace 6 units to the positive x axis is the second shape. Center is the morphing shape. The transformation node is also used for drawing the labels shape1,2 and morphing.

Since linear morphing is required, the general linear interpolation equation, p = p1 + (p2-p1)*t is used to create the linear interpolation model for transformation. Where t is the time function. So the x1,y1,z1 function of the first function will morph to the second x2,y2,z2 function linearly.

However this function only morphs the function from shape 1 to 2 and will not morph the reverse way. This can be easily achieved by adding another modulus function 1-abs(1-2*t). and p = p1 + (p2-p1)*back_and_forth(t) instead of *t.

This is done in file: animation_F&B.wrl

Rendering

Cycle Interval of 7 is used for slower transformation. Loop set to True so that is will keep morph forever. Resolution used for all three shapes is 80 80 80 for the best rendering speed(less than 2 seconds) but most smooth appearance and jitter free animation as tested on my computer.