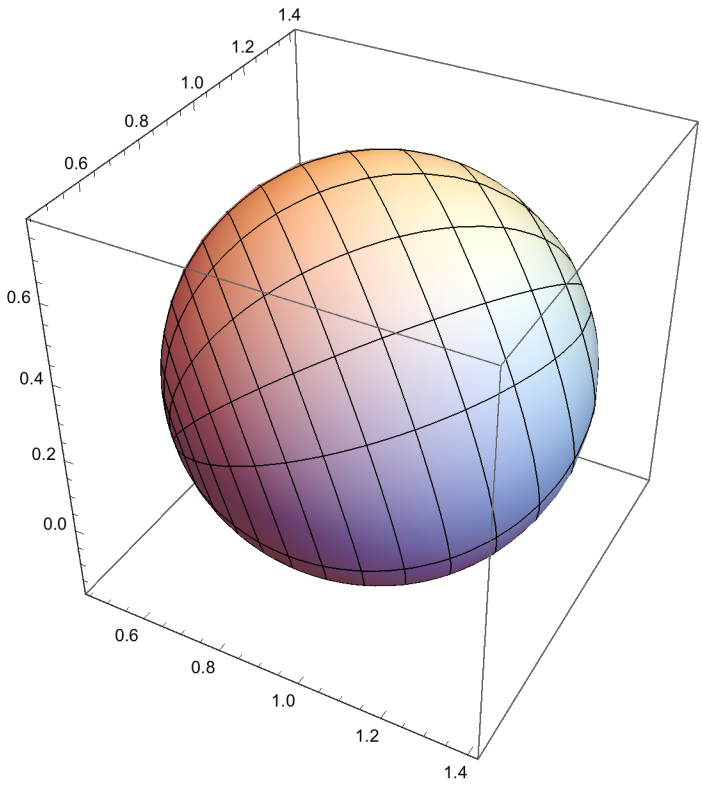
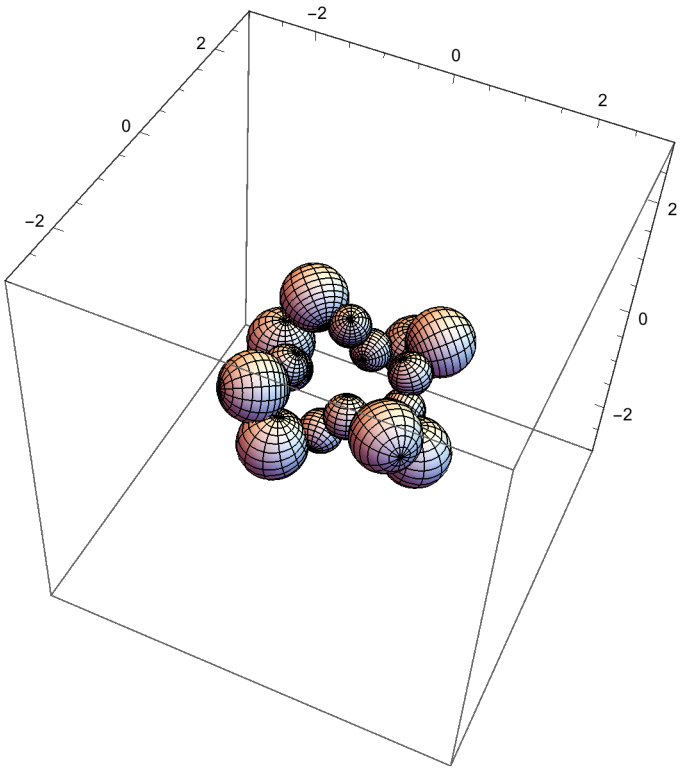
```
en  = 1;
(* en = 2; *)
n = 16;
p = 4;
tau = (p - en) * 2 * Pi / n;
temp = ((Sin[2 * Pi * p / n]) ^2  * (Cos[2 * Pi / n] - Cos[tau])) /
    ((Sin[2 * Pi / n]) ^2 * (Cos[tau] - Cos[2 * Pi * p / n]));
epsilon = Sqrt[1 / (temp + 1)];
 r = 2 * Pi / 18;  (*For en=1, n=16, p=4*)
(*   r = 2*Pi/13;    For en=2, n=16, p=4*)

(*Start with an initial point (x0,y0,u0,v0) on our three-sphere,
such that x0^2 + y0^2 + u0^2 +v0^2 = 1, and radius r *)
(*
x0 = 1/Sqrt[2];
y0 = 1/Sqrt[2];
u0 = 0;
v0 = 0;
*)
x0 = Sqrt[(1 - epsilon^2) / 2];
y0 = Sqrt[(1 - epsilon^2) / 2];
u0 = epsilon / (Sqrt[2]);
v0 = epsilon / (Sqrt[2]);
(*Start with an initial point (x0,y0,u0,v0) on our three-sphere,
such that x0^2 + y0^2 + u0^2 +v0^2 = 1, and radius r *)
(*x0 = 1/Sqrt[2];
y0 = 1/Sqrt[2];
u0 = 0;
v0 = 0;
r = Pi/8;
n=8;
p=1;*)

(* Rotation: Rotation about the uv-
  plane to get our point out of the x-axis. i.e. set the x component to
   be zero. Then define a rotation function fuv using the angle t1 *)
```

```
If[y0 == 0, t1 = Pi / 2, t1 = ArcTan[x0 / y0]];
fuv =
  {{Cos[t1], -Sin[t1], 0, 0}, {Sin[t1], Cos[t1], 0, 0}, {0, 0, 1, 0}, {0, 0, 0, 1}};
(* Initial point *)
p0 = {{x0}, {y0}, {u0}, {v0}};
(* p1: Point in the yuv-plane. i.e. the x component is zero *)
p1 = fuv.p0;
(* Rotate about the xv-plane to get our point out of the y-
 axis. i.e. set the y component to be zero. Then
  define a rotation function fxv using the angle t2 *)
y1 = p1[[2, 1]];
u1 = p1[[3, 1]];
If[u1 == 0, t2 = Pi / 2, t2 = ArcTan[y1 / u1]];
fxv =
  {{1, 0, 0, 0}, {0, Cos[t2], -Sin[t2], 0}, {0, Sin[t2], Cos[t2], 0}, {0, 0, 0, 1}};
(* p2: Point in the uv-plane. i.e. the y component is zero *)
p2 = fxv.p1;
(* Rotate about the xy-plane to get our point out of the u-
 axis. i.e. set the u component to be zero. Then
  define a rotation function fxy using the angle t3 *)
u2 = p2[[3, 1]];
v2 = p2[[4, 1]];
If[v2 == 0, t3 = Pi / 2, t3 = ArcTan[u2 / v2]];
fxy =
  {{1, 0, 0, 0}, {0, 1, 0, 0}, {0, 0, Cos[t3], -Sin[t3]}, {0, 0, Sin[t3], Cos[t3]}};
(* p3: Point in the v-axis. i.e. the u component is zero *)
p3 = fxy.p2;

(* Parametrization: Now we parametrize out new point with ease,
since it lies on the v-axis *)
(* r is the radius of our three-sphere,
from the initial values. theta and phi are the parametrization variables *)
x[theta_, phi_] := Sin[r] * Sin[phi] * Cos[theta]
y[theta_, phi_] := Sin[r] * Sin[phi] * Sin[theta]
u[theta_, phi_] := Sin[r] * Cos[phi]
v[theta_, phi_] := Cos[r]
(* Next, we apply the inverse of our rotation functions
 to return our point to its initial location on S^3 *)
paraPoint = Inverse[fxy.fxv.fuv].
   {{x[theta, phi]}, {y[theta, phi]}, {u[theta, phi]}, {v[theta, phi]}};

(* Our new vector has four, non-zero, components. All dependent on phi *)
```

```
(* Next, we apply the stereographic projection
  to our parametrized vector and plot it to the xyu-
 plane. The stereographic projection is a mapping from S^3 to R^3 *)

xPrime = ((paraPoint[[1, 1]]) / (1 - paraPoint[[4, 1]]));
yPrime = ((paraPoint[[2, 1]]) / (1 - paraPoint[[4, 1]]));
uPrime = ((paraPoint[[3, 1]]) / (1 - paraPoint[[4, 1]]));

ParametricPlot3D[{xPrime, yPrime, uPrime}, {theta, 0, 2 * Pi}, {phi, 0, Pi}]
```



```
isom = {{Cos[2 * Pi / n], -Sin[2 * Pi / n], 0, 0},
    {Sin[2 * Pi / n], Cos[2 * Pi / n], 0, 0}, {0, 0, Cos[2 * Pi * p / n], -Sin[2 * Pi * p / n]},
    {0, 0, Sin[2 * Pi * p / n], Cos[2 * Pi * p / n]}};
Show[Table[newpt = MatrixPower[isom, i].paraPoint;
  newxprime = newpt[[1, 1]] / (1 - newpt[[4, 1]]);
  newyprime = newpt[[2, 1]] / (1 - newpt[[4, 1]]);
  newuprime = newpt[[3, 1]] / (1 - newpt[[4, 1]]);
  ParametricPlot3D[{newxprime, newyprime, newuprime},
   {theta, 0, 2 * Pi}, {phi, 0, Pi}, PlotRange → {-3, 3}], {i, 1, n}]]
```



```
z0 = x0 + I * y0;
w0 = u0 + I * v0;
curve[t_] :=
  {{Cos[t], -Sin[t], 0, 0}, {Sin[t], Cos[t], 0, 0}, {0, 0, Cos[p * t], -Sin[p * t]},
    {0, 0, Sin[p * t], Cos[p * t]}}.{{x0}, {y0}, {u0}, {v0}};
x = (curve[t][[1, 1]] / (1 - curve[t][[4, 1]]));
y = (curve[t][[2, 1]] / (1 - curve[t][[4, 1]]));
z = (curve[t][[3, 1]] / (1 - curve[t][[4, 1]]));
Show[ParametricPlot3D[{x, y, z}, {t, 0, 2 * Pi}, PlotRange → {-2, 2}],
 ParametricPlot3D[{xPrime, yPrime, uPrime},
  {theta, 0, 2 * Pi}, {phi, 0, Pi}, PlotRange → {-2, 2}]]
```