```
In[1]:= n = 8;
NextBulbNormal[𝕔_][{{x_, y_, z_}, Jz_: {}}] := Block[{r, w, c, s, 𝕫, 𝕣},

  r = Sqrt[x^2 + y^2 + z^2];
  w = Sqrt[x^2 + y^2];
  cθ = x/w;
  sθ = y/w;
  cϕ = z/r;
  sϕ = w/r;
  𝕫θ = (cθ + 𝕚 sθ)^n;
  𝕫ϕ = (cϕ + 𝕚 sϕ)^n;
  c̲θ = Re[𝕫θ];
  s̲θ = Im[𝕫θ];
  c̲ϕ = Re[𝕫ϕ];
  s̲ϕ = Im[𝕫ϕ];
  𝕣 = r^n;
  z = 𝕣 {c̲θ s̲ϕ, s̲θ s̲ϕ, c̲ϕ} + 𝕔;

  (*Note early out*)
  If[Jz == {}, Return[{z}]];
  (*Note early out*)

  𝔸 = ( c̲θ s̲ϕ   𝕣 s̲ϕ   0       0   𝕣 c̲θ
        s̲θ s̲ϕ   0       𝕣 s̲ϕ   0   𝕣 s̲θ
        c̲ϕ       0       0       𝕣   0    );

  M = MatrixPower;

  𝔹 = ( n r^(n-1)   0                              0
        0           n M[(cθ -sθ; sθ cθ), n - 1]    0
        0           0                              n M[(cϕ -sϕ; sϕ cϕ), n - 1] ) // ArrayFlatten;


  ℂ = ( 1        0        0     0     0
        0        -x/w^2   1/w   0     0
        0        -y/w^2   0     1/w   0
        -z/r^2   0        0     0     1/r
        -w/r^2   1/r      0     0     0   );

  𝔻 = ( x/r   y/r   z/r
        x/w   y/w   0
        1     0     0
        0     1     0
        0     0     1   );

  𝕁𝕫 = 𝔸.𝔹.ℂ.𝔻.Jz + IdentityMatrix[3];
  {z, 𝕁𝕫}
 ]

MandelBulbNormal[𝕔_] :=
 Nest[NextBulbNormal[𝕔], N@{𝕔, IdentityMatrix[3]}, recursionLevel] /. {z_, 𝕁𝕫_} :> z.𝕁𝕫 //
  Normalize

recursionLevel = 2;
MandelBulb[𝕔_] := Norm@Nest[NextBulbNormal[N@𝕔], N@{𝕔}, recursionLevel]
```