Pensieve header: Planet Earth pulled back to the Hopf fibration- programs.

```
size = 2400;
SetOptions[Rasterize, {RasterSize → size, ImageSize → size}];
SetOptions[Graphics, ImageSize → size];
SetOptions[Graphics3D, {
    ImageSize → size, Boxed → False, AspectRatio → 1,
    ViewVector → {vp = 4 {1, 0.1, 1}, {-1, -0.3, -1.5}},
    ViewAngle → 38 Degree,
    Lighting → {{"Point", White, vp}, {"Point", White, {0, 0, 0}}},
    Background → Black
  }];
SetAttributes[MakeImage, HoldRest];
MakeImage[name_String, g_] := Timing[Module[{s},
    Export[name <> ".png", s = ImageCrop[Rasterize[img[name] = g]]];
    ImageResize[s, 512]
  ]]

(AllCountries = Union[Take[CountryData["Countries"], All], {"Antarctica"}; {}]);
G20 = {"Argentina", "Australia", "Brazil", "Canada",
    "China", "France", "Germany", "India", "Indonesia", "Italy",
    "Japan", "Mexico", "Russia", "SaudiArabia", "SouthAfrica",
    "SouthKorea", "Turkey", "UnitedKingdom", "UnitedStates"};
G8 = CountryData["G8"];

MakeColourScheme[ac_List, newborders_List] := Module[
    {c, nbd, cnbd, addborders, k, oc, nc, changes, change},
    (nbd[#] = CountryData[#, "BorderingCountries"]) & /@ ac;
    addborders[{c1_, c2_}] := (
      nbd[c1] = Union[nbd[c1], {c2}];
      nbd[c2] = Union[nbd[c2], {c1}];
     );
    addborders[l_List] := Do[
      addborders[{l[[i]], l[[j]]}],
      {i, 2, Length[l]}, {j, i}
     ];
    addborders /@ newborders;
    (CountryColour[#] = RandomReal[]) & /@ AllCountries;
    changes = {}; change = 0;
    Do[
     Do[
      cnbd = Sort[CountryColour /@ nbd[c]];
      If[cnbd ≠ {},
       AppendTo[cnbd, 1 + First[cnbd]];
       {k} = Ordering[Rest[RotateRight[cnbd] - cnbd], 1];
       oc = CountryColour[c];
       nc = FractionalPart[(cnbd[[k]] + cnbd[[k + 1]]) / 2];
       oc = Abs[oc - nc]; If[oc > 0.5, oc = 1 - oc];
       change += oc;
       CountryColour[c] = nc
```

```
      ],
      {c, AllCountries}
     ];
     AppendTo[changes, change];
     change = 0,
     {32}
    ];
    Print["Country colouring process changes: ", changes]
  ];
MakeColourScheme[ac_] := MakeColourScheme[ac, {}];
LoadColourScheme[s_String] := Module[
   {rule},
   rule = Get["ColourSchemes/" <> s <> ".m"];
   (CountryColour[#] = (# /. rule)) & /@ (First /@ rule);
  ];
SaveColourScheme[] := SaveColourScheme[
   DateString[{"YearShort", "Month", "Day", "-", "Hour24", "Minute", "Second"}]
  ];
SaveColourScheme[dt_String] := Module[
  {},
  Put[
   (# → CountryColour[#]) & /@ AllCountries,
   "ColourSchemes/" <> dt <> ".m"
  ];
  Export["ColourSchemes/" <> dt <> ".png",
   Graphics[{
     EdgeForm[White],
     {
        Hue[CountryColour[#]],
        CountryShape[#] /. {CountryShape → List,
          Region[pts___] :→ Polygon[{pts} /. LonLat → List]}
       } & /@
      (AllCountries)
    },
    Background → Black, ImageSize → 640
   ]
  ]
 ]

Area[reg_Region] := Abs[Dot[
    Sin[List @@ Last /@ reg],
    ((RotateLeft[#] - #) &) @ (List @@ First /@ reg)
   ]];
Area[c_CountryShape] := (Plus @@ Area /@ c);
Area[s_String] := Area[CountryShape[s]];
```

A "CountryShape" is a sequence of "Region"s, each of which is a sequence of "Point"s:

```
CountryShape[c_String] := Sort[
  Select[
   Map[(LonLat @@ N[# Degree]) &,
    Apply[Region,
     CountryShape @@ First[CountryData[c, "SchematicPolygon"]],
     {1}
    ],
    {2}
   ],
   Area[#] > 10^(-5) &
  ],
  Area[#] &
 ]
(* /. LonLat[ln_, lt_] :> LonLat[-Pi/2+ln, lt] *)

beta = 0;
pi[C2[z_, w_]] := Module[
   {x1 = Re[z], x2 = Im[z], x3 = Re[w], x4 = Im[w]},
   R[2 (x1 x3 + x2 x4), 2 (x2 x3 - x1 x4), x1^2 + x2^2 - x3^2 - x4^2]
  ];
sigma[LonLat[phi_, theta_]] :=
  C2[Exp[-I (beta + phi)] * Sqrt[(1 + Sin[theta]) / 2], Sqrt[(1 - Sin[theta]) / 2]];
tau[LonLat[phi_, theta_]] := C2[Sqrt[(1 + Sin[theta]) / 2],
   Exp[I (beta + phi)] * Sqrt[(1 - Sin[theta]) / 2]];
S1Act[a_, C2[z_, w_]] := C2[E^(I a) * z, E^(I a) * w];
lambda[C2[z_, w_]] := R3 @@ ({Re[z], Im[z], Re[w]} / (1 - Im[w]));

Options[Whirl] = {
   WhirlBottom → Pi / 4, WhirlTop → 7 Pi / 4, WhirlingSteps → 24,
   WhirlRegions → All,
   WhirlStyle → {},
   BottomStyle → {},
   TopStyle → {},
   Section → tau
  };
Whirl[r_Region, opts___] := Module[
   {
    whirlstart = WhirlBottom /. {opts} /. Options[Whirl],
    whirlend = WhirlTop /. {opts} /. Options[Whirl],
    whirlingsteps = WhirlingSteps /. {opts} /. Options[Whirl],
    whirlstyle = WhirlStyle /. {opts} /. Options[Whirl],
    bottomstyle = BottomStyle /. {opts} /. Options[Whirl],
    topstyle = TopStyle /. {opts} /. Options[Whirl],
    section = Section /. {opts} /. Options[Whirl],
    reg, angles, mesh, out , arc
   },
   reg = section /@ r;
   reg = Append[List @@ reg, First[reg]];
   angles = N[Table[
      a,
```

```
        {a, whirlstart, whirlend, (whirlend - whirlstart) / (whirlingsteps /. 0 → 1)}
       ]];
     mesh = Outer[(List @@ lambda[S1Act[#1, #2]]) &, angles, reg];
     out = {
       Opacity[1], EdgeForm[Black],
       Glow[], bottomstyle, Polygon[Rest[First[mesh]]],
       Opacity[1], EdgeForm[Black], Glow[], topstyle, Polygon[Rest[Last[mesh]]]
      };
     If[whirlingsteps > 0, AppendTo[out,
       {
        Opacity[1], EdgeForm[], Glow[], whirlstyle,
        Table[
         Polygon[
          {mesh[[i, j]], mesh[[i + 1, j]], mesh[[i + 1, j + 1]], mesh[[i, j + 1]]}],
         {i, Length[angles] - 1}, {j, Length[reg] - 1}
        ]
       }
      ]];
     Flatten[out]
    ];
  Whirl[c_String, opts___] := Module[
    {whirlregions = WhirlRegions /. {opts} /. Options[Whirl]},
    {
     Hue[CountryColour[c]],
     Take[List @@ CountryShape[c], whirlregions] /. r_Region :→ Whirl[r, opts]
    }
   ]
```

```
grid := {
   Blue,
   Table[
    arc = Table[
      tau[LonLat[lon Degree // N, lat Degree // N]],
      {lat, -90, 80, 2}
     ];
    {
     Line[(List @@ lambda[S1Act[40 Degree, #]]) & /@ arc],
     Line[(List @@ lambda[S1Act[-40 Degree, #]]) & /@ arc]
    },
    {lon, -180, 160, 20}
   ],
   Table[
    arc = Table[
      tau[LonLat[lon Degree // N, lat Degree // N]],
      {lon, -180, 180, 1}
     ];
    {
     Line[(List @@ lambda[S1Act[40 Degree, #]]) & /@ arc],
     Line[(List @@ lambda[S1Act[-40 Degree, #]]) & /@ arc]
    },
    {lat, -90, 80, 10}
   ]
  };
```