

Copyright March 26, 2006. doug@youvan.com www.youvan.com www.pseudocolor.com

```

pp3d =
  ParametricPlot3D[{(Sin[2*Pi*ang/360]+1)*50, (Sin[2*Pi*2*ang/360]+1)*50,
    (Cos[2*Pi*ang/360]+1)*50}, {ang, 1, 360}, ImageSize -> {250, 250},
  AxesLabel -> {RED, GREEN, BLUE}, ViewPoint -> {0.05, 0., 2.}];
Export["C:\\Documents and Settings\\K64S\\Desktop\\Mathematica\\Lissij\\1.gif",
  pp3d, "GIF", ImageSize -> {250, 250}];

draw = Table[{x, y}, {x, 1, 150}, {y, 1, 150}];
draw[[All, All]] = {0, 0, 0};

drawxdown = Table[{x, y}, {x, 1, 150}, {y, 1, 150}];
drawxdown[[All, All]] = {0, 0, 0};

drawshift = Table[{x, y}, {x, 1, 150}, {y, 1, 150}];
drawshift[[All, All]] = {0, 0, 0};

drawshiftxdown = Table[{x, y}, {x, 1, 150}, {y, 1, 150}];
drawshiftxdown[[All, All]] = {0, 0, 0};

Lissij = Table[{x}, {x, 1, 3600}];
Lissij[[All]] = {0, 0, 0};

(* Lissij is a list of 3600 elements (0.1 degrees),
  each with an r,g,b = x,y,z value (respectively) on a scale of 1-100;
  the numbers 1.03 and 49 are being used to stop Floor from creating a zero
  value and to keep all values less than or equal to 100. 2*Pi is used for
  degrees rather than radians. One revolution, 360 d , is calculated. *)

For[ang = 1, ang <= 3600, ang++,
  Lissij[[ang]] = {Floor[(Sin[2*Pi*ang/3600]+1.03)*49], Floor[
    (Sin[2*Pi*2*ang/3600]+1.03)*49], Floor[(Cos[2*Pi*ang/3600]+1.03)*49]}];

(* Below, two values of 25 each are used to raise and right-
  shift the image within the black background so it is centered *)

For[i = 1, i <= 3600, i++, {a, b, c} = Lissij[[i]];
  If[1 <= a <= 100 && 1 <= b <= 100, draw[[a+25, b+25]] = Lissij[[i]/100]];

(* matrix element 1,1 is upper left,
  whereas it is lower left in an image, so swap matrix top to
  bottom for rows using drawxdown as an intermediate then reassign*)

For[i = 1, i < 150, i++, x = 151 - i; drawxdown[[i, All]] = draw[[x, All]];
draw[[All, All]] = drawxdown[[All, All]];

gdraw = Graphics[RasterArray[Apply[RGBColor, draw, {2}]],
  AspectRatio -> error, ImageSize -> {200, 200}];
Show[gdraw];
Export["C:\\Documents and Settings\\K64S\\Desktop\\Mathematica\\Lissij\\2.gif",
  gdraw, "GIF", ImageSize -> {200, 200}];

```

```

(* black filler for stereo pair *)
side = Table[{x, y}, {x, 1, 150}, {y, 1, 100}];
side[[All, All]] = {0, 0, 0};

(* make stereo drawing 600 pixels wide *)
stereo = Table[{x, y}, {x, 1, 150}, {y, 1, 600}];
stereo[[All, All]] = {0., 0., 0.};

(* 'draw' is loaded twice to check with no stereo shift *)

For[i = 1, i ≤ 150, i++, For[j = 1, j ≤ 600, j++,
  If[j ≤ 100, stereo[[i, j]] = side[[i, j]]];
  If[100 < j ≤ 250, stereo[[i, j]] = draw[[i, j - 100]]];
  If[250 < j ≤ 350, stereo[[i, j]] = side[[i, j - 250]]];
  If[350 < j ≤ 500, stereo[[i, j]] = draw[[i, j - 350]]];
  If[500 < j ≤ 600, stereo[[i, j]] = side[[i, j - 500]]]; Continue
]]

gstereo = Graphics[RasterArray[Apply[RGBColor, stereo, {2}],
  AspectRatio → error, ImageSize → {600, 150}];
Show[gstereo];
Export["C:\\Documents and Settings\\K64S\\Desktop\\Mathematica\\Lissij\\3.gif",
  gstereo, "GIF", ImageSize → {600, 150}];

(* move (y=green channel) pixels left
  ( proportional to the height of the z=blue channel) to pop z=
  blue up out of the page, i.e., upward in Schrodinger cube,
  away from bottom x=red, y=green plane *)

(* variable 'draw' has xy coordinates from 1-150 and rgb values from 0-1;
  shifted image goes into 'drawshift' *)

For[i = 1, i ≤ 150, i++, For[j = 1, j ≤ 150, j++, {a, b, c} = draw[[i, j]]; a = Floor[100 * a];
  b = Floor[100 * b]; c = Floor[100 * c]; anew = a; bnew = Floor[b - (0.1 * c)];
  cnew = c; drawshift[[anew + 25, bnew + 25]] = {anew / 100, bnew / 100, cnew / 100}];

(* this next inversion of rows is necessary for unknown
  reason! The step above inverts drawshift relative to draw - why? *)

For[i = 1, i < 150, i++, x = 151 - i; drawshiftdown[[i, All]] = drawshift[[x, All]];
drawshift[[All, All]] = drawshiftdown[[All, All]];
drawshiftdown =.

(* rebuild 'stereo' with shifted image on right *)

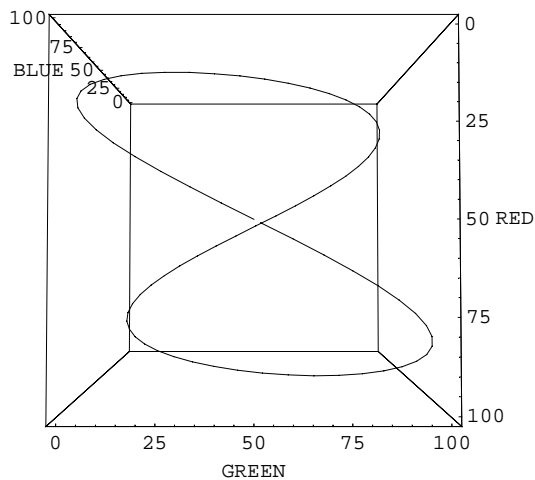
stereo[[All, All]] = {0, 0, 0};

For[i = 1, i ≤ 150, i++, For[j = 1, j ≤ 600, j++,
  If[j ≤ 100, stereo[[i, j]] = side[[i, j]]];
  If[100 < j ≤ 250, stereo[[i, j]] = draw[[i, j - 100]]];
  If[250 < j ≤ 350, stereo[[i, j]] = side[[i, j - 250]]];
  If[350 < j ≤ 500, stereo[[i, j]] = drawshift[[i, j - 350]]];
  If[500 < j ≤ 600, stereo[[i, j]] = side[[i, j - 500]]]]];

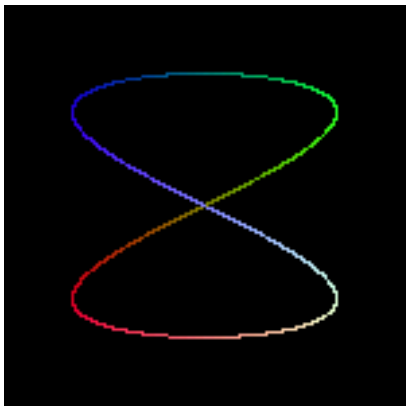
```

```
(* error is invoked in AspectRatio so, after recovery,
we get proper resolution, size and aspect ratio - this is a BUG *)

gstereo = Graphics[RasterArray[Apply[RGBColor, stereo, {2}]],
  AspectRatio -> error, ImageSize -> {600, 150}];
Show[gstereo];
Export["C:\\Documents and Settings\\K64S\\Desktop\\Mathematica\\Lissij\\4.gif",
  gstereo, "GIF", ImageSize -> {600, 150}];
```



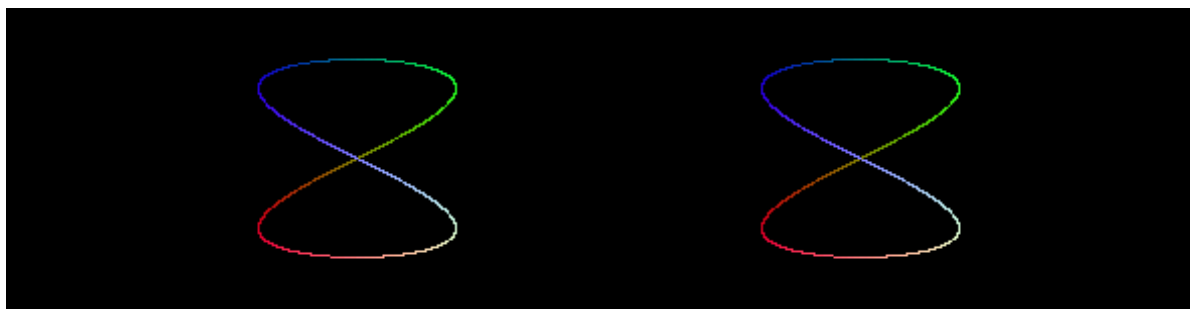
```
AspectRatio::aspr :
Value of option AspectRatio -> error is not a finite positive number or Automatic. More...
```



```
AspectRatio::aspr :
Value of option AspectRatio -> error is not a finite positive number or Automatic. More...
```

```
AspectRatio::aspr :
Value of option AspectRatio -> error is not a finite positive number or Automatic. More...
```

```
AspectRatio::aspr :
Value of option AspectRatio -> error is not a finite positive number or Automatic. More...
```



```

AspectRatio::aspr :
  Value of option AspectRatio -> error is not a finite positive number or Automatic. More...

AspectRatio::aspr :
  Value of option AspectRatio -> error is not a finite positive number or Automatic. More...

AspectRatio::aspr :
  Value of option AspectRatio -> error is not a finite positive number or Automatic. More...

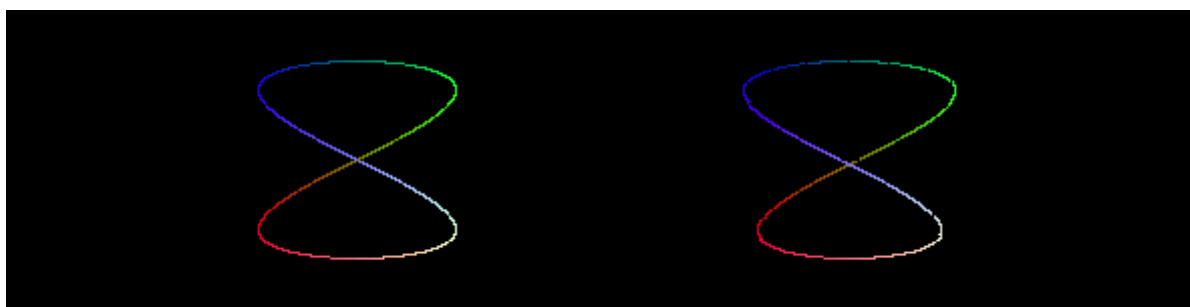
Graphics::realu : Argument in RGBColor[ $\frac{87}{100}, -\frac{1}{100}, \frac{17}{100}$ ] is not a real number between 0 and 1. More...

Graphics::realu : Argument in RGBColor[ $\frac{43}{50}, -\frac{1}{100}, \frac{4}{25}$ ] is not a real number between 0 and 1. More...

Graphics::realu : Argument in RGBColor[ $\frac{17}{20}, -\frac{1}{100}, \frac{3}{20}$ ] is not a real number between 0 and 1. More...

General::stop : Further output of Graphics::realu will be suppressed during this calculation. More...

```



```

AspectRatio::aspr :
  Value of option AspectRatio -> error is not a finite positive number or Automatic. More...

AspectRatio::aspr :
  Value of option AspectRatio -> error is not a finite positive number or Automatic. More...

Graphics::realu : Argument in RGBColor[ $\frac{87}{100}, -\frac{1}{100}, \frac{17}{100}$ ] is not a real number between 0 and 1. More...

Graphics::realu : Argument in RGBColor[ $\frac{43}{50}, -\frac{1}{100}, \frac{4}{25}$ ] is not a real number between 0 and 1. More...

Graphics::realu : Argument in RGBColor[ $\frac{17}{20}, -\frac{1}{100}, \frac{3}{20}$ ] is not a real number between 0 and 1. More...

General::stop : Further output of Graphics::realu will be suppressed during this calculation. More...

```