# Lava Lamp

**A cultural icon of the 1960s is resurrected in digital form. Functions to generate a lava lamp animation are presented, and an alternate embodiment of lava is discussed.**

*Robert Rudd*

"*The guy who keeps staring at the lava lamps is back again.*"
— conversation between clerks at a Spencer Gifts store

This note presents functions to generate an animation sequence that resembles a lava lamp. In a lava lamp, heat provided by the lamp causes density changes in the lava. As the lava heats, a buoyancy force stretches the lava in the vertical direction until it breaks into two parts. One part continues moving upward while the other part cools and sinks to the bottom. After breaking apart, both pieces undergo a radial oscillation. Since modeling the physics of the lava is a formidable task, we rely on observation to construct the graphics.

### The Lamp

The lamp consists of four separate graphics objects: a top, a bottom, an end cap, and glass. It is rotationally symmetric about the vertical axis, so we will use cylindrical coordinates to describe it.

```
In[1]:=  cylinder[r_, phi_, z_] := {r Cos[phi], r Sin[phi], z}
```

The radius $r$ is given by the following function of height $z$, a sinusoid with a frequency chirp.

```
In[2]:=  r[z_] := 2 - 0.7 Sin[Pi z(1 - z^2/512)/4] // N
```

The base and top are formed by wrapping this parametrized curve about the vertical axis.

```
In[3]:=  base = ParametricPlot3D[
             Evaluate[cylinder[r[z], phi, z]],
             {phi, 0, 2 Pi}, {z, 0, 4},
             PlotPoints -> {17, 17},
             DisplayFunction -> Identity];
```

```
In[4]:=  top = ParametricPlot3D[
             Evaluate[cylinder[r[z], phi, z]],
             {phi, 0, 2 Pi}, {z, 11, 12},
             PlotPoints -> {17, 4},
             DisplayFunction -> Identity];
```

*Robert Rudd works for BF Goodrich Aerospace in Vergennes, Vermont. He received Bachelors and Masters Degrees in mechanical engineering from Rensselaer Polytechnic Institute in Troy, New York, and has worked at the Unisys and Singer Corporations.*

The end cap is a disk.

```
In[5]:=  cap = ParametricPlot3D[
             Evaluate[cylinder[r, phi, 12]],
             {r, 0, r[12]}, {phi, 0, 2 Pi},
             PlotPoints -> {2, 17},
             DisplayFunction -> Identity];
```
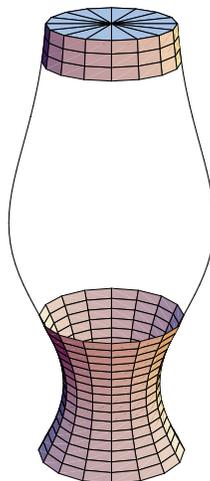
The glass is implied by the two curves corresponding to $\phi = 0$ and $\phi = \pi$.

```
In[6]:=  glass = ParametricPlot3D[
             Evaluate[cylinder[r[z], #, z]& /@ {0, Pi} ],
             {z, 4, 11},
             PlotPoints -> 25,
             DisplayFunction -> Identity];
```

The four components are assembled into an object called `lamp`.

```
In[7]:=  lamp = Show[{top, base, glass, cap},
                Boxed -> False, Axes -> False];
```

```
In[8]:=  Show[lamp,
             ViewPoint -> {0, 4, 1.5},
             DisplayFunction -> $DisplayFunction]
```

## The Lava

There are four pieces of lava in the animation, placed at equal radii, in steps of $\pi/2$ about the vertical axis. Each lava piece undergoes a sinusoidal translation in the vertical direction, one quarter cycle out of phase from the others. Each piece starts with an approximately spherical shape. It stretches in the vertical direction until it breaks into two parts. Both parts oscillate radially as one rises and the other sinks.

One cycle of the lava runs from zero to one in time. The critical time at which the lava splits is denoted tc.

```
In[9]:=  tc = 0.25;
```

Each lava piece consists of two halves which share an edge until they break apart. To describe the shape of the parts, we switch to spherical coordinates.

```
In[10]:=  sphere[r_, phi_, theta_] :=
            r {Cos[phi]Sin[theta], Sin[phi]Sin[theta], Cos[theta]}
```

The radius of the lava is given by spherical harmonic functions. The initial radius is:

```
In[11]:=  r0[theta_] := 0.8 - 0.2 LegendreP[2, Cos[theta]] // N
```

The radius is constant in time prior to the critical time and an exponentially damped sinusoid after.

```
In[12]:=  pc[theta_] := 0.3 LegendreQ[3, Cos[(theta + .003)/2]] -
                        0.25 LegendreP[1, Cos[theta]] // N
```

The time dependence is given by a conditional function:

```
In[13]:=  f[t_/;(t < tc)] =  1;
```

```
In[14]:=  f[t_/;(t >= tc)] :=  Sqrt[2] *
            Exp[-3(t - tc)] Cos[6 Pi(t-tc) + Pi/4] // N
```

```
In[15]:=  rc[theta_, t_] := 0.8(1 + pc[theta] f[t])
```

The initial shape will be transformed smoothly into the time-dependent shape by a simple "morph":

```
In[16]:=  morph[f1_, f2_, s_] := s f1 + (1 - s) f2
```

Because the lava pieces share an edge until the critical time, they must be translated vertically. The vertical distance is constant until the critical time and decays exponentially after the critical time.

```
In[17]:=  g[t_/;(t < tc)] = 1;
```

```
In[18]:=  g[t_/;(t >= tc)] := Exp[-10(t - tc)]
```

```
In[19]:=  rc0 = rc[0, tc];
```

Here are the initial and time-dependent lava shapes, given as parametrized surfaces. The vertical offset is added to the time-dependent function.

```
In[20]:=  lava0[phi_, theta_] :=
            sphere[r0[(Pi + theta)/2], phi, (Pi + theta)/2] // N
```

```
In[21]:=  lavac[phi_, theta_, t_] :=
            sphere[rc[theta, t], phi, theta] - {0, 0, rc0 g[t]} // N
```

The morphing of one function into the other is parametrized by a Gaussian function of time. The function pgauss is extended by periodicity to provide continuity at the start and end of the animation cycle. The second argument of pgauss specifies the standard deviation.
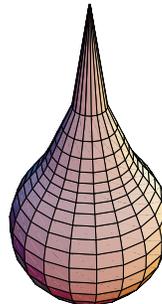
```
In[22]:=  pgauss[t_, sig_] :=
            Exp[-(Mod[t, 1]/sig)^2] + Exp[-((Mod[t, 1] - 1)/sig)^2]
```

The formula for half a lava piece is:

```
In[23]:=  lavahalf[phi_, theta_, t_] :=
            0.9 morph[lava0[phi, theta], lavac[phi, theta, t],
                pgauss[t, 0.11]]
```

At the critical time, the lava has a cusp at $\theta = 0$:

```
In[24]:=  ParametricPlot3D[
            Evaluate[lavahalf[phi, theta, tc]],
            {phi, 0, 2 Pi}, {theta, 0, Pi},
            Boxed -> False, Axes -> False]
```



To provide a more continuous look, the parametrization is modified by replacing $\theta$ with a quadratic function of $\theta$:

```
In[25]:=  biastheta[theta_] := theta(1 + 2 theta/Pi)/3
```

The derivative of this function at $\theta = 0$ is 1/3, so the effect is to triple the polygon density near the cusp at the expense of the polygon density around $\theta = \pi$.

```
In[26]:=  ParametricPlot3D[
          Evaluate[lavahalf[phi, biastheta[theta], tc]],
          {phi, 0, 2 Pi}, {theta, 0, Pi},
          Boxed -> False, Axes -> False]
```
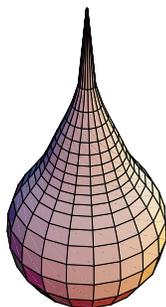


Here is a plot of half a lava piece at various stages in its evolution:

```
In[27]:=  Show[GraphicsArray[
          ParametricPlot3D[
          Evaluate[
            lavahalf[phi, biastheta[theta], #] ],
            {phi, 0, 2 Pi}, {theta, 0, Pi},
            Boxed -> False, Axes -> False,
            DisplayFunction -> Identity]& /@
          {0, 0.2, 0.4} ],
          DisplayFunction -> $DisplayFunction]
```



The lava parts move in the vertical direction. The upper part moves sinusoidally; the lower part follows the upper part until the critical time and then sinks to the bottom.

```
In[28]:=  hu[t_] := 7.0 - 3.8 Cos[2 Pi t] // N
```

```
In[29]:=  {c1, c2} = {hu[tc] + hu[0], hu[tc] - hu[0]}/2;
```

```
In[30]:=  hl[t_/;(t < tc)] := hu[t]
```

```
In[31]:=  hl[t_/;(t >= tc)] :=
              c1 + c2 Cos[Pi(t - tc)/(1 - tc)] // N
```

```
In[32]:=  Plot[{hu[t], hl[t]}, {t, 0, 1}]
```
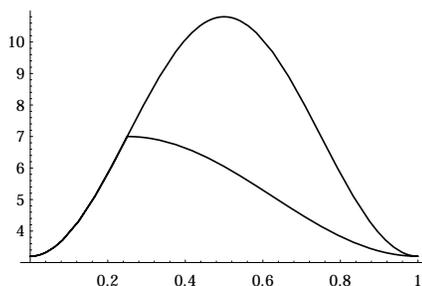


The function lava generates the lower part, reflects it in the $x$-$y$ plane to create the upper part, and translates both parts into their vertical positions.

```
In[33]:=  Needs["Graphics`Shapes`"]
```

```
In[34]:=  lava[t_] := Module[{blob},
              blob = ParametricPlot3D[Evaluate[
                  lavahalf[phi, biastheta[theta], t]],
                  {phi, 0, 2 Pi}, {theta, 0, Pi},
                  PlotPoints -> {13, 17},
                  DisplayFunction -> Identity];
              {TranslateShape[blob, {0, 0, hl[t]}],
               TranslateShape[AffineShape[blob, {1, 1, -1}],
                   {0, 0, hu[t]}]} ]
```

There are four lava pieces in the animation, arranged about the vertical axis at equal radii. The pieces are out of phase by one quarter of a cycle. To put the $k$-th piece out of phase, the time parameter is replaced by this function of $t$ and $k$:

```
In[35]:=  modt[t_, k_] := Mod[t - k/4, 1]
```

The function lava4 creates the four lava pieces and places them about the vertical axis.

```
In[36]:=  p[phi_] := {Cos[phi], Sin[phi], 0}
```

```
In[37]:=  pk[k_] := 0.65 p[Pi/8 + k Pi/2]
```

```
In[38]:=  lava4[t_] := Show[
              Table[
                  TranslateShape[lava[modt[t, k]], pk[k]],
                  {k, 4}] ]
```

### Colors

The color of the lamp is gold, to suggest a gold anodized aluminum finish. Blue is chosen for the lava color since gold has no blue in it allowing independent color control with the lighting. The colors for the graphics objects and lighting come from the standard package Graphics`Colors`.

```
In[39]:=  Needs["Graphics`Colors`"]
```

The function editcolor inserts a SurfaceColor directive into a Graphics3D object.

```
In[40]:=  editcolor[object_, diffuse_, spec_, n_] :=
              Insert[object, SurfaceColor[diffuse, spec, n], {1, 1}]
```

In this function, diffuse and spec are the diffuse and specular reflection colors, and n is the exponent of the specular reflection.

```
In[41]:=  goldlamp = editcolor[lamp, Gold, GrayLevel[0], 4];
```

The function showLamp assembles the lamp and a given configuration of lava into one frame of the animation.

```
In[42]:=  showLamp[lava_] := Show[
            editcolor[lava,
              RGBColor[0, 0, .65], RGBColor[.75, 0, 1], 6],
            goldlamp,
            LightSources ->
              {{{1, 2, 1}, RGBColor[1, 0, 0]},
               {{1, 1, 1}, RGBColor[0,.5, 0]},
               {{0, 1, 1}, RGBColor[0, 0, 1]},
               {{1, 0, 1}, Gold},
               {{.5, 10, 1}, Blue},
               {{.5,-10, 1}, Blue}},
            Boxed -> False, Axes -> False,
            ViewPoint -> {0, 4, 1.5},
            PlotRange -> {{-3, 3}, {-3, 3}, {0, 12.01}},
            DisplayFunction -> $DisplayFunction]
```
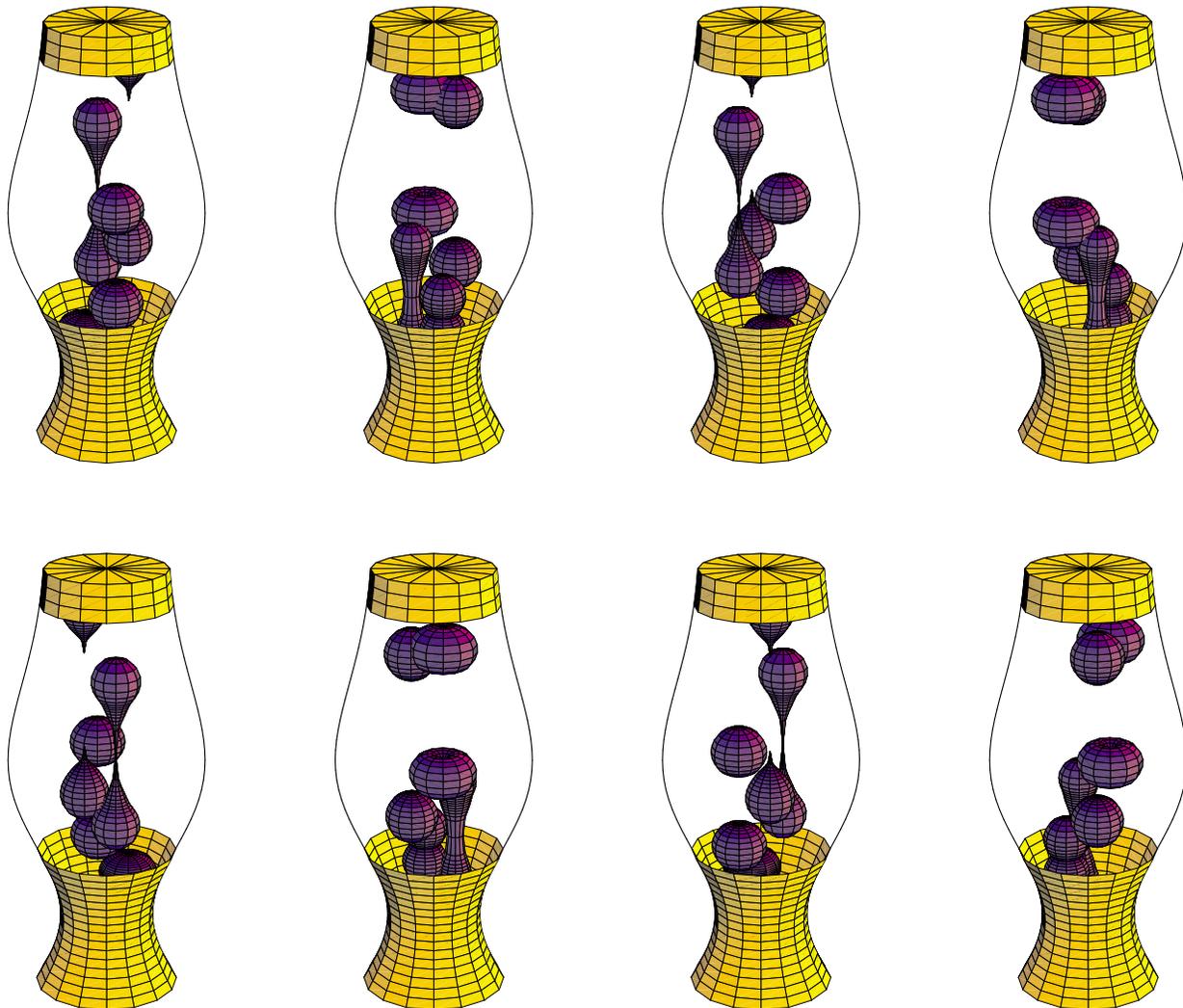
The function `lavalamp` generates an animation sequence with $n$ frames:

```
In[43]:=  lavalamp[n_] := Table[showLamp[lava4[t/n]], {t, 0, n-1}]
```

Thirty six frames give a smooth animation.

On machines with limited memory, the number of plot points in the lava pieces can be reduced, or the number of lava pieces can be reduced to three or two. Another type of lava which requires less memory is described below.

## Alternate Lava

One of the nice features of computer graphics is that it is not constrained by reality. If we give up our attempts at realism, we find that polyhedra can make nice looking lava.

```
In[44]:=  Needs["Graphics`Polyhedra`"]

In[45]:=  poly =
            {Polyhedron[Octahedron,   {0, 0, 0}, 0.60 ],
             Polyhedron[Dodecahedron,{0, 0, 0}, 0.70 ],
             Polyhedron[Icosahedron,  {0, 0, 0}, 0.75 ],
             Polyhedron[Hexahedron,   {0, 0, 0}, 0.65 ]};
```

The function `placelava` positions the polyhedra.

```
In[46]:=  placelava[lava_, trans_, rot_] :=
            TranslateShape[
              RotateShape[lava, rot, rot, 0], trans]
```

The rotation angle of the $k$-th polyhedron is

In[47]:= `twist[t_, k_] := Pi/k Sin[Pi k t/2]`

A reservoir is provided at the bottom of the lamp for the polyhedra to emerge from.

In[48]:=
```
reservoir[t_] :=
    ParametricPlot3D[Evaluate[
      rl p[phi] + {0, 0, 1} *
      (3.9 + Cos[Pi/2 rl/r[3.9]] *
        Sum[2 pgauss[t - k/4 - 0.125, 0.125] *
          Exp[-4 #.#&@(rl p[phi] - pk[k])],
          {k, 4}])],
      {rl, 0, r[3.8]}, {phi, 0, 2 Pi},
      PlotPoints -> {17, 33},
      DisplayFunction -> Identity]
```

The four polyhedra and the reservoir are assembled in the function `polylava`.

In[49]:=
```
polylava[t_] :=
    Show[ reservoir[t],
      Table[
        placelava[ poly[[k]],
          pk[k] + {0, 0, hu[modt[t, k]]},
          twist[t, k]],
        {k, 4}],
      DisplayFunction -> Identity]
```

In[50]:= `showLamp[polylava[0.3]]`
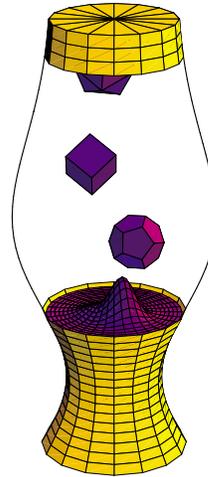


The function `polylavalamp` generates an animation sequence with $n$ frames.

In[51]:=
```
polylavalamp[n_] :=
    Table[showLamp[polylava[t/n]], {t, 0, n-1}]
```

Robert Rudd
B.F. Goodrich, Panton Road, Vergennes, VT05491

The electronic supplement contains the notebook `Lava Lamp`.