Geometrica for *Mathematica* 6

Bruno Autin
MuSoftware
bruno.autin@mac.com

Calling *Geometrica*

Each time you need *Geometrica*, execute the following command.

```
<< Geometrica`
```

To see the results of the commands contained in that notebook and in any other notebook of the *Geometrica* documentation, execute the input cells sequentially. If you forget to do so, you may have to deal with undefined variables.

Geometrica versus Cabri or Sketchpad

Features	Cabri or Sketchpad	Geometrica
interactivity	mouse	mouse, keyboard or palette
geometry	euclidean	analytical
functions	"**"	"****"
multiple objects	no	yes
abstract objects	no	yes
geometrical functions	no	yes

Geometrica and *Mathematica*

list processing
symbolic facilities
functional programming
Draw and Draw3D commands
many new geometrical functions

Euclidean and Analytical geometries

Primary concept in EUCLIDEAN geometry: CPoint → superb reasoning (unfortunately disappearing!)

Primary concept in ANALYTICAL geometry: COORDINATES → unification of geometry and algebra (Descartes and the theory of curves).

Solving an elementary problem

The best navigation consists of choosing a simple problem and to see how it is solved. We shall study the generation of a parabola as the envelope of one edge of a square ruler whose vertex describes a line and the other edge passes through a fixed point.

▪ Point, line and first drawing

```
m = CPoint[1, 0]
```

```
CPoint[1, 0]
```

The straight line is the y-axis

```
d = CLine[1, 0, 0]
```

```
CLine[1, 0, 0]
```

▪ Bound point and euclidean line

The edge passing through m is defined by a second point bound to d:

```
m1 = Pointer[d, t]
d1 = ELine[m, m1]
```

```
CPoint[0, -t]
```

```
CLine[t, -1, -t]
```

▪ Geometrical function

The other edge d_2 is perpendicular in m_1 to d_1.

```
d2 = ELine[m1, d1]
```

```
CLine[1, t, t^2]
```

- ■ **Derivative of a geometrical function**

When the parameter *t* varies, the line d_2 moves.Between two infinitesimally close positions, the two lines intersect at a point of the envelope.

```
d21 = D[#, t] & /@ d2
mp = d2 ⋂ d21
```

```
CLine[0, 1, 2 t]
```

```
CPoint[t², -2 t]
```

- ■ **Viewing the envelope process**

We are now curious to see the motion of the line d_2 or, more precisely of the square ruler (m,m1,mp). This is an opportunity to show a geometrical animation using the Manipulate function of *Mathematica* 6.0.

First, the fixed elements of the diagram are drawn. Let us note that mp is nothing but the parametric representation of a curve. In *Geometrica*, a curve is described by the function PPoint whose arguments are the two coordinates written as pure functions and the range of variation (rule PRange) of the parameter:

```
p = PPoint[#^2 &, -2 # &, PRange -> {-5, 5}];
```

Text is added to point m and line d using the function Legend which is Listable. The option Offset→{r,φ} shifts the text with respect to the object on a cicle of radius r at an azimuth φ.

```
te = Legend[{"m", "d"}, {m, d}, Offset -> {1.5, 0}];
```

Then the graph is produced but not displayed by terminating the statement with a semi-colon.

```
g1 = Draw[DrawRange[-5, 5], te, Blue, m, d, p];
```

The variable part of the diagram is the square ruler defined by

```
sr[t_] = Segment[m, m1, mp];
```

In *Geometrica*, the function Segment is used not only for an ordinary segment of straight line limited by two points but for any polygonal line. The animation is produced by the display of the fixed (g1) and variable (g2) diagrams using Manipulate which has the same syntax as Table.

```
Manipulate[g2 = Draw[Red, sr[t]];
 Show[g1, g2], {t, -5, 5}]
```
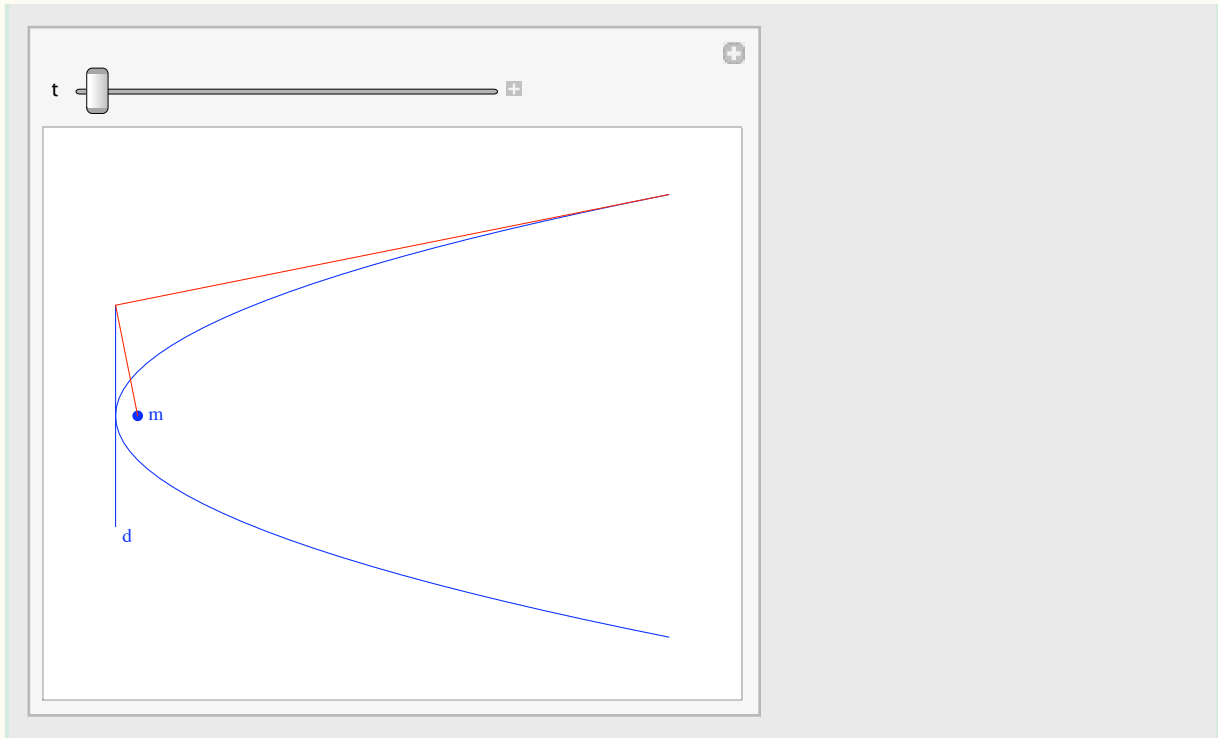
Illustration in 3D

The previous problem can be visualized in 3D by replacing the parabola by a cylinder with a parabolic directrix.

▪ 3D conic

A 2D object can be converted into a 3D object using the command `To3D`.

```
p3D = To3D[p]
```

$$PPoint\left[\#1^2 \&, -2 \#1 \&, 0 \#1 \&, PRange \to \{-5, 5\}\right]$$

The parabola is defined by its 2D representation and located in the horizontal plane. A point belonging to the parabola is still obtained using `Pointer`,

```
t = .;
m3D = Pointer[p3D, t]
```

$$CPoint\left[t^2, -2 t, 0\right]$$

and the tangents using `ELine`.

```
tg = ELine[p3D, t]
```

$\text{CLine}\left[\text{CPlane}[0, 0, 1, 0], \text{CPlane}\left[-2, -2\,t, 0, -(-2 - 2\,t)\,t^2 - 2\,t\,\left(2\,t + t^2\right)\right]\right]$

- **Cylinder**

A cylinder is defined by a curve, here the parabola, and a generator parallel to a fixed direction *Oa* where *a* has been chosen as the point of *Oz* of height 1. The cylinder is given in parametric form. The 2D parabola *p* is automatically interpreted as a 3D curve in `ECylinder`.

```
cyp = ECylinder[CPoint[0, 0, 1], p]
```

$\text{PPoint}\left[\#1^2\,\&, -2\,\#1\,\&, \#2\,\&, \text{PRange} \to \{\{-5, 5\}, \{-\pi, \pi\}, \{25, 2\}\}\right]$

The second parameter describes a point on a generator. By default, the height varies between $-\pi$ and π. Let us change that range to (-10,10).

```
cyp = cyp /. _Rule → (PRange → {{-5, 5}, {-10, 10}, {25, 2}})
```

$\text{PPoint}\left[\#1^2\,\&, -2\,\#1\,\&, \#2\,\&, \text{PRange} \to \{\{-5, 5\}, \{-10, 10\}, \{25, 2\}\}\right]$

The planes tangent to the parabolic cylinder are represented by parallelograms. A parallelogram is a special quadrangle that may be defined by three vertices only. The first vertex, called here a1, is the origin of the parallelogram. It is deduced from the origin a of the tangent to the parabola by translation of vector (0,0,-10). The second vertex is obtained by translation of the contact point m3D by the vector (0,0,-10. The third point a2 is symmetric to a1 with respect to the reference horizontal plane.

```
a = Pointer[tg, 0];
{a1, a2} = Translate[a, CPoint[0, 0, {-10, 10}]]
```

$\left\{\text{CPoint}\left[-\dfrac{t^2}{1+t^2}, -\dfrac{t^3}{1+t^2}, -10\right], \text{CPoint}\left[-\dfrac{t^2}{1+t^2}, -\dfrac{t^3}{1+t^2}, 10\right]\right\}$

```
b = Translate[m3D, CPoint[0, 0, -10]]
```

$\text{CPoint}\left[t^2, -2\,t, -10\right]$

The parallelogram is then given by

```
Clear[pp]
pp[t_] = Parallelogram[a1, b, a2]
```

$$\text{Segment}\left[\text{CPoint}\left[-\frac{t^2}{1+t^2}, -\frac{t^3}{1+t^2}, -10\right], \text{CPoint}\left[t^2, -2t, -10\right],\right.$$

$$\left.\text{CPoint}\left[t^2, -2t, 10\right], \text{CPoint}\left[-\frac{t^2}{1+t^2}, -\frac{t^3}{1+t^2}, 10\right], \text{CPoint}\left[-\frac{t^2}{1+t^2}, -\frac{t^3}{1+t^2}, -10\right]\right]$$

The full figure can now be drawn:
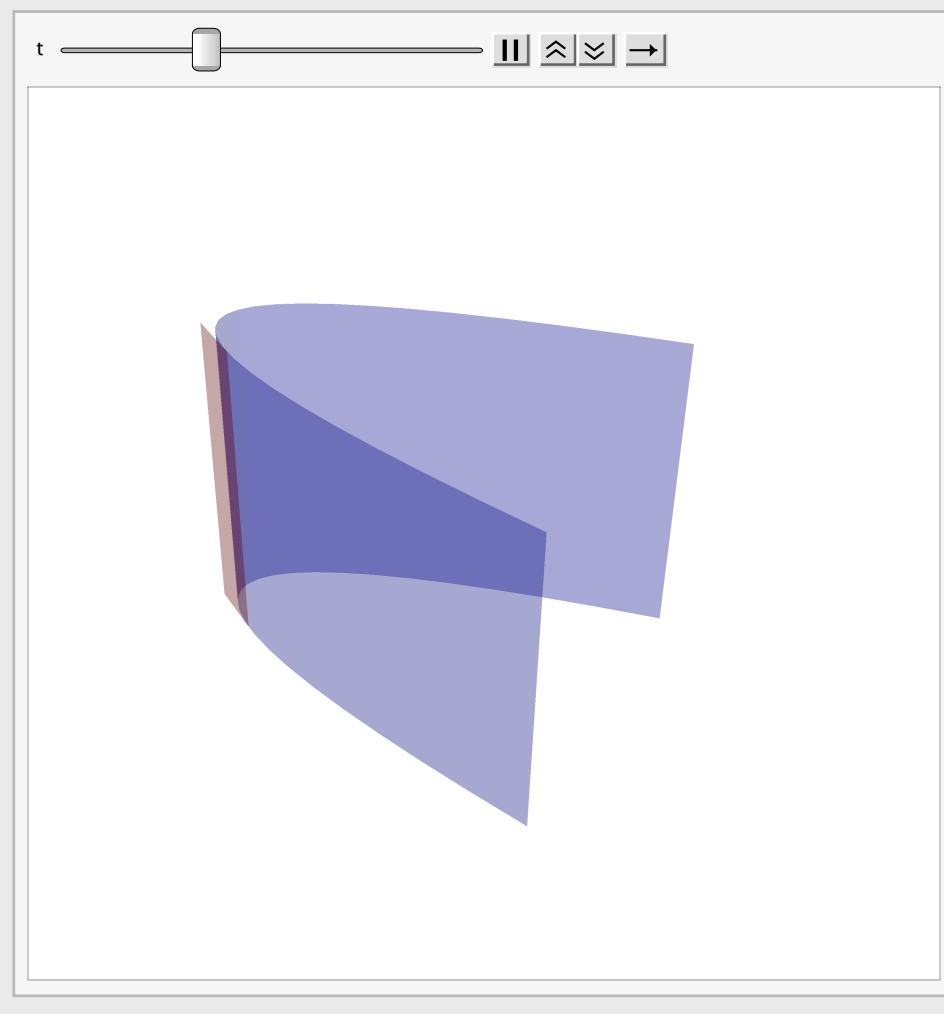
```
g3 = Draw3D@Paint[cyp, Blue];
```

```
Animate[g4 = Draw3D@Paint[pp[t], Red];
 Show[g3, g4],
 {t, -5, 5}]
```

The colors are not as vivid as one might expect because the objects are transparent. The graphics directive Opacity is set to .4 by default in *Geometrica*. In the next example, its is set to 1 for the parallelograms.
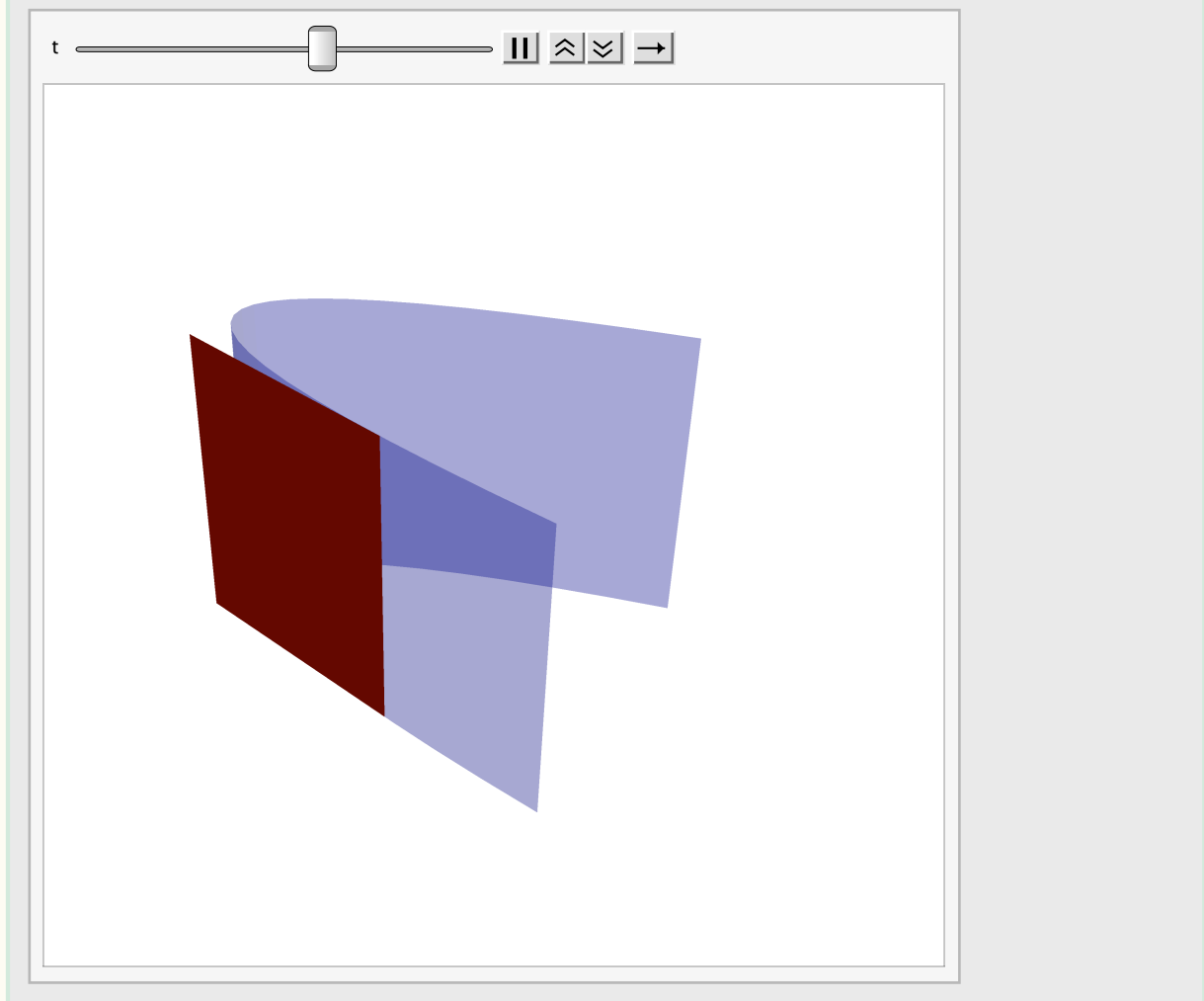
Tour of Geometrica

```
Animate[g4 = Draw3D[Opacity[1], Paint[pp[t], Red]];
  Show[g3, g4],
  {t, -5, 5}]
```

Tour of Geometrica

Tour of Geometrica

CAD functions

Tour of Geometrica

▪ Walls as examples of parallel polygons

Tour of Geometrica

A line *L'* is parallel to a line *L* when it is generated by the end of a segment attached and normal to *L*. This defintion assumes that *L* represents a continuous and derivable function, which is not true for a polygon which has no normal at a vertex. A convention is introduced in *Geometrica* to define a parallel polygon. It is illustrated for the walls of a house.
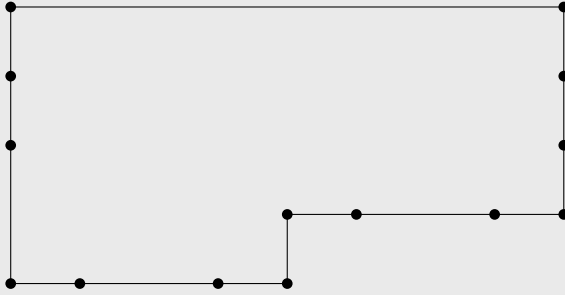
```
x = {0, 1, 3, 4, 4, 5, 7, 8, 8, 8, 8, 0, 0, 0};
y = {0, 0, 0, 0, 1, 1, 1, 1, 2, 3, 4, 4, 3, 2};
m = CPoint[x, y];
s = Close[Segment @@ m];
Draw[m, s]
```

The wall is a list of polygons. Some polygons may be windows or doors. They can be identified by varying the color of the polygons. In this example, there are one door (in gray) and three windows (in white).

```
w = Wall[s, .3];
col = {Black, White, Black, Black, Black, White,
    Black, Black, White, Black, Black, Black, Gray, Black};
Draw[Paint[w, col]]
```

- **Putting dimensions on a technical drawing**

The drawing can finally be conpleted with the dimensions of the wall.

```
dim = Dimension[s];
Draw[Paint[w, col], dim]
```
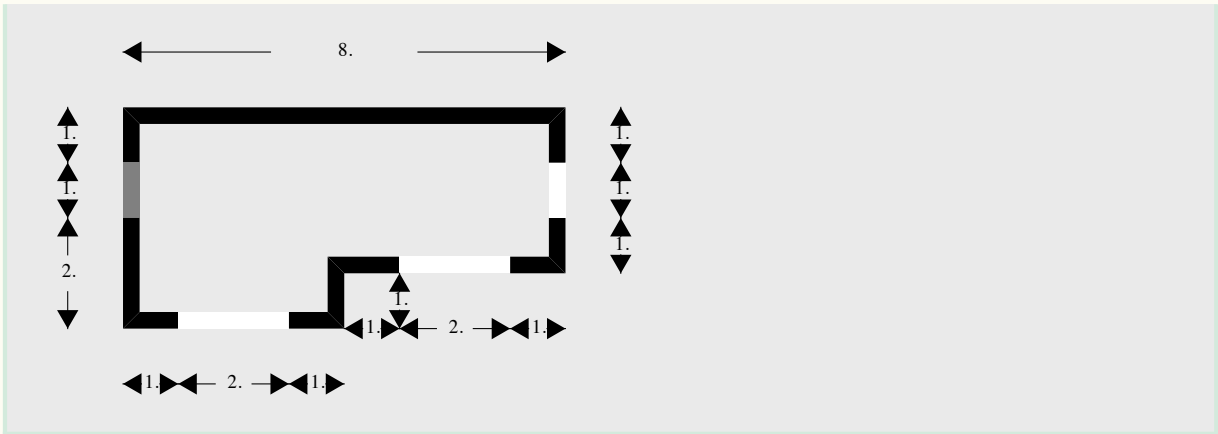
- **Helicoidal staircase**
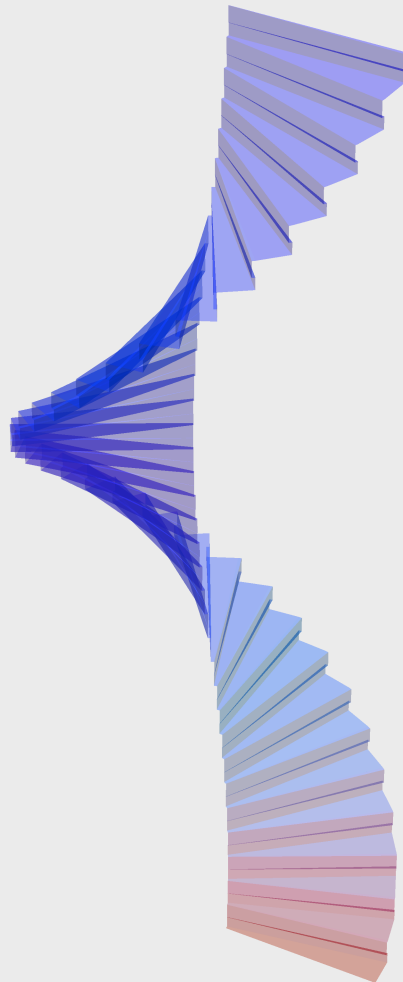
To have a first view of a helicoidal staircase, one uses the default definition.

```
sc = HelicoidalStaircase[];
Draw3D@Paint[sc]
```

Then various variables and options are at hand to parametrize the staircase. The detail of a step is shown by drawing $\frac{1}{36}$ of the full staircase whose stair width is equal to 1, pitch is 1 and angular step is $\frac{\pi}{18}$. Each stair has a thickness and a recess which have optional values. Optional are also the sense of rotation, the inner radius and the initial angle. The stair is represented here from two opposite view points.
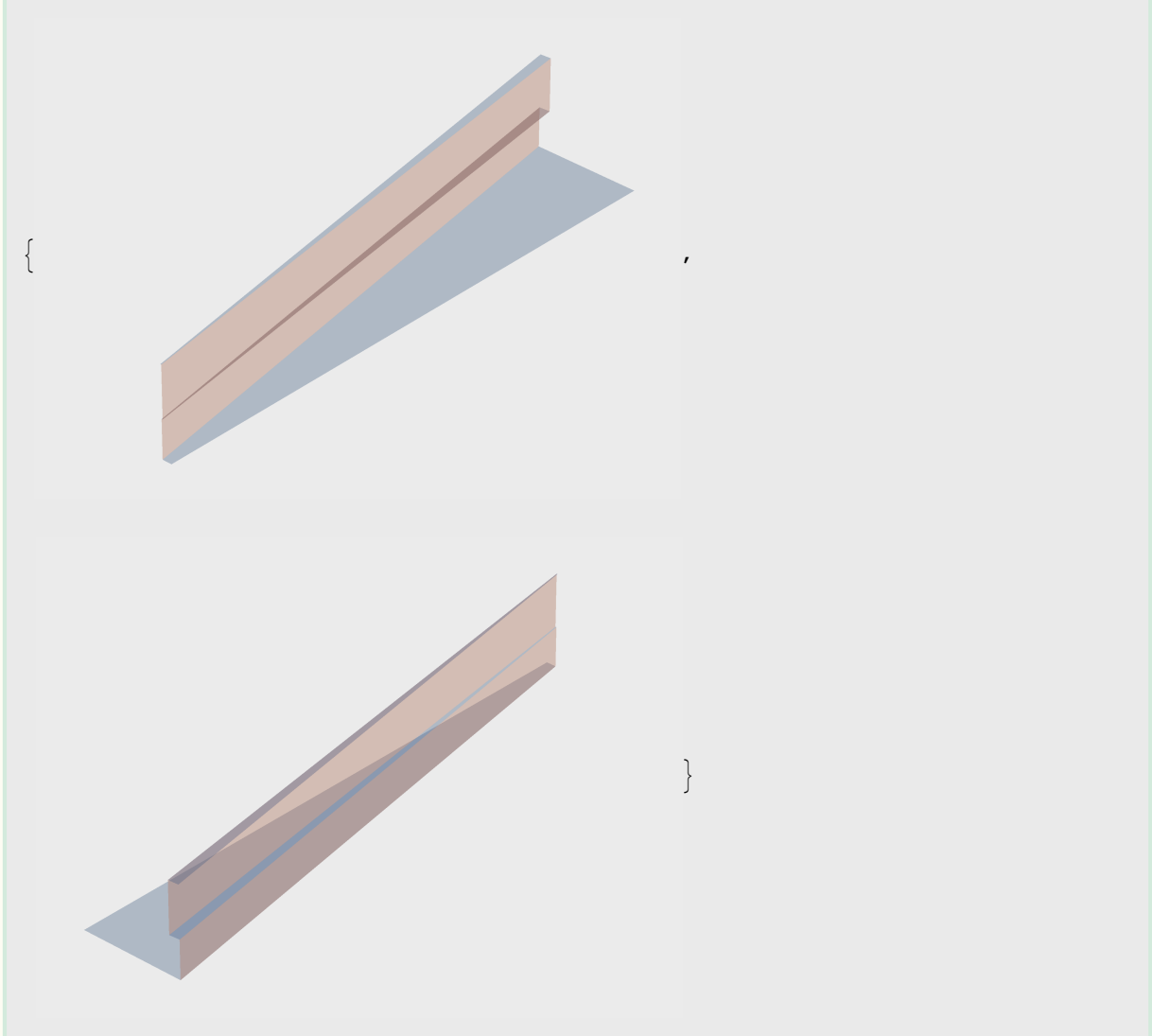
```
sc = HelicoidalStaircase[];
Draw3D@Paint[sc]
```

```
stair = HelicoidalStaircase[1, 1, 1 / 36, Pi / 18];
Draw3D[Paint[List @@ stair, {GrayLevel[.4], GrayLevel[.6],
        GrayLevel[.4], GrayLevel[.6], GrayLevel[.4]}], #] & /@
  {ViewPoint → {-5, -5, 5}, ViewPoint → {5, 5, 5}}
```

Variations on parallelism

The function `Parallel` in *Geometrica* is common in CAD applications where it is rather called `Offset`. A parallel to a curve or surface C is generated by a point n located on the normal at C in m and at distance d from m. In *Geometrica*, the distance d may be fixed or variable. The simplest case concerns the straight line whose parallel is also a straight line.

```
d = CLine[0, 1, 0];
d1 = Parallel[d, .5];
Draw[Blue, d, Red, d1]
```
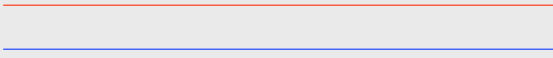
The notion of parallelism is however much more general than that simple case. The next diagram represents a particle oscilla-
tion of fixed amplitude about a circle. Here the distance varies and the reference curve is a circle. The replacement (50 → 200)
improves the definition of the curve by acting on the third argument of PRange.

```
{m1, m2} = CPoint[{2, 1}, 0];
c = ECircle[m1, m2]
l = Parallel[c, .1 Sin[10 #] &] /. (50 → 200)
Draw[Blue, c, Red, l]
```

```
Ellipse[1, 0, 1, -2, 0, 3]
```
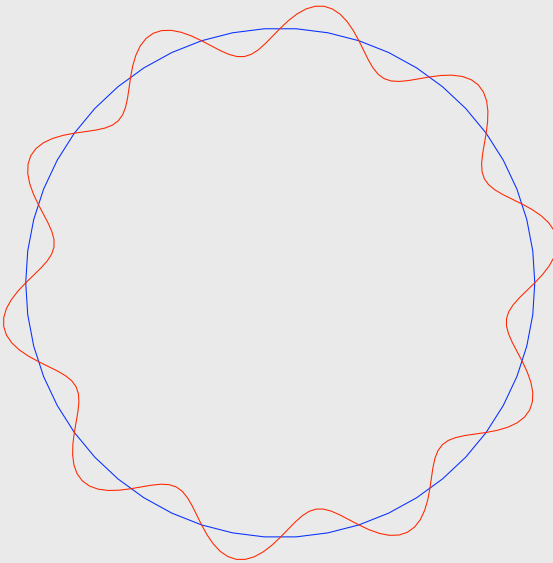
```
PPoint[2 + Cos[#1] (1. + 0.1 Sin[10 #1]) &, Sin[#1] (1. + 0.1 Sin[10 #1]) &, PRange → {-π, π, 200}]
```

The curve may take fancier shapes for larger amplitudes.

```
l = Parallel[c, Sin[10 #] &] /. (50 → 200);
Draw[Blue, c, Paint[l, Red], PlotRange -> All]
```

```
l = Parallel[c, 2 Sin[10 #] &] /. (50 → 400);
Draw[Blue, c, Paint[l, Red], PlotRange -> All]
```

The function *Parallel* applies to any 3D curve or surface as well. Next comes a parallel curve of helix.

```
l = Parallel[c, Sin[10 #] &] /. (50 → 200);
Draw[Blue, c, Paint[l, Red], PlotRange -> All]
```

Tour of Geometrica

```
h = Helix[];
l = Parallel[h, 2 Sin[#] &] /. (24 → 100)
Draw3D[Blue, h, Red, l]
```

Tour of Geometrica

```
PPoint[Cos[#1] (1 - 2 Sin[#1]) &, -1 + Cos[2 #1] + Sin[#1] &, #1 &, PRange → {0., 6.28319, 100}]
```

Tour of Geometrica

Tour of Geometrica

Parallel surfaces can also be obtained by modulating one or both parameters. This is a cylinder whose elliptic base is deformed by a sine function.

Tour of Geometrica

```
s = EllipticCylinder[];
fu = Sin[#1] &;
sp = Parallel[s, fu]
Draw3D@Paint[sp, Blue]
```
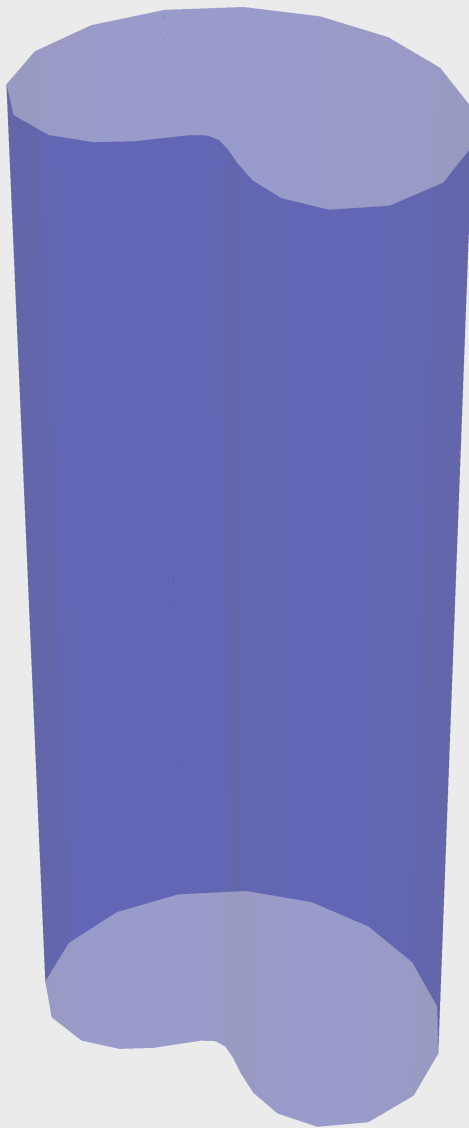
Tour of Geometrica

$$\text{PPoint}\Big[\text{Cos}[\#1] + \frac{54\,608\,393\,\text{Cos}[\#1]\,\text{Sin}[\#1]}{\sqrt{2\,982\,076\,586\,042\,449\,\text{Cos}[\#1]^2 + 5\,964\,153\,172\,084\,900\,\text{Sin}[\#1]^2}}\; \&,$$

$$\frac{54\,608\,393\,\text{Sin}[\#1]}{77\,227\,930} + \frac{\text{Sin}[\#1]^2}{\sqrt{\frac{2\,982\,076\,586\,042\,449\,\text{Cos}[\#1]^2}{5\,964\,153\,172\,084\,900} + \text{Sin}[\#1]^2}}\; \&,$$

$$\#2\; \&,\; \text{PRange} \to \{\{-\pi,\ \pi\},\ \{-\pi,\ \pi\},\ \{25,\ 2\}\}\Big]$$

Tour of Geometrica

$$\text{PPoint}\Big[\text{Cos}[\#1] + \frac{54\,608\,393\,\text{Cos}[\#1]\,\text{Sin}[\#1]}{\sqrt{2\,982\,076\,586\,042\,449\,\text{Cos}[\#1]^2 + 5\,964\,153\,172\,084\,900\,\text{Sin}[\#1]^2}}\; \&,$$

$$\frac{54\,608\,393\,\text{Sin}[\#1]}{77\,227\,930} + \frac{\text{Sin}[\#1]^2}{\sqrt{\frac{2\,982\,076\,586\,042\,449\,\text{Cos}[\#1]^2}{5\,964\,153\,172\,084\,900} + \text{Sin}[\#1]^2}}\; \&,$$

The generators only are now modified using the second variable. As the generator is deformed by the modulation of the second parameter, the parameter range is redefined to give the same definition fo both types of isocurves.
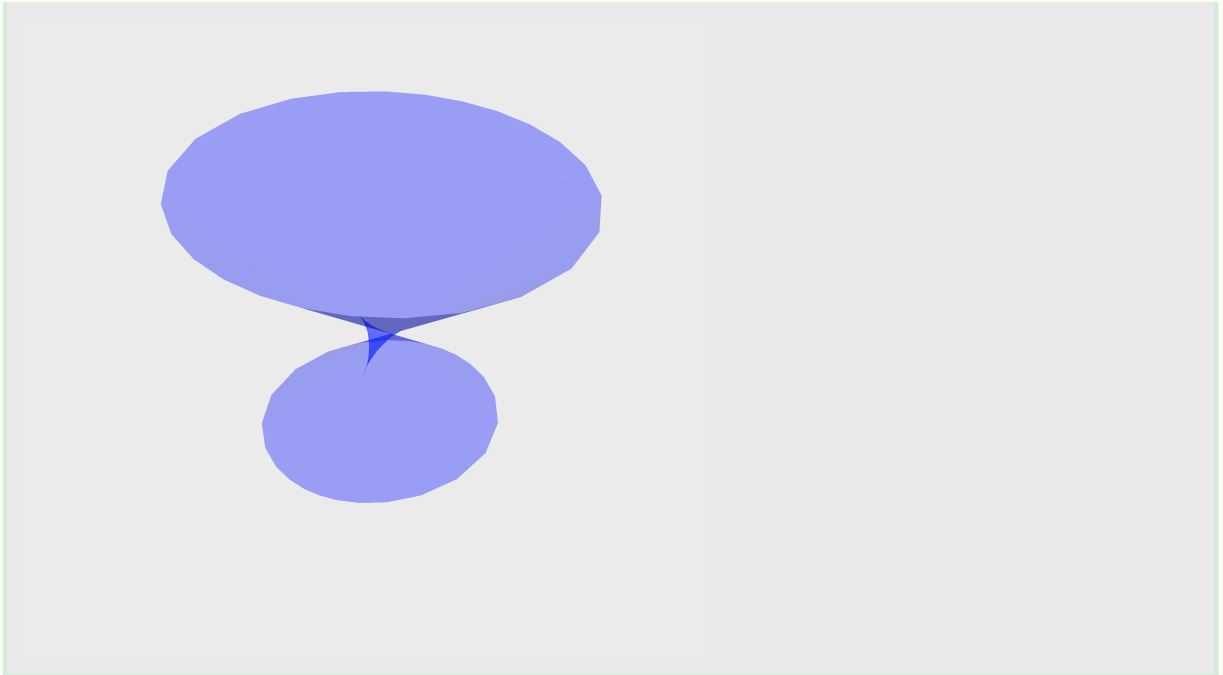
```
fv = 1.2 #2 &;
sp = Parallel[s, fv] /.
   _Rule → (PRange → {{-π, π}, {-π, π}, {25, 25}})
Draw3D@Paint[sp, Blue]
```

$$\text{PPoint}\Big[\text{Cos}[\#1] + \frac{6.55301\times10^{7}\,\text{Cos}[\#1]\,\#2}{\sqrt{2\,982\,076\,586\,042\,449\,\text{Cos}[\#1]^{2} + 5\,964\,153\,172\,084\,900\,\text{Sin}[\#1]^{2}}}\,\&,$$

$$\frac{54\,608\,393\,\text{Sin}[\#1]}{77\,227\,930} + \frac{1.2\,\text{Sin}[\#1]\,\#2}{\sqrt{\frac{2\,982\,076\,586\,042\,449\,\text{Cos}[\#1]^{2}}{5\,964\,153\,172\,084\,900} + \text{Sin}[\#1]^{2}}}\,\&,$$

$$\#2\,\&,\ \text{PRange} \to \{\{-\pi, \pi\}, \{-\pi, \pi\}, \{25, 25\}\}\Big]$$

The original cylinder is now submitted to the product of the two previous paraxial functions. The surface can be manipulated by clicking the figure and make it rotate to get different view points and understand how it is folded.

```
fuv = 1.2 #2 Sin[#1] &;
sp = Parallel[s, fuv] /.
   _Rule → (PRange → {{-π, π}, {-π, π}, {25, 25}})
Draw3D@Paint[sp, Blue]
```

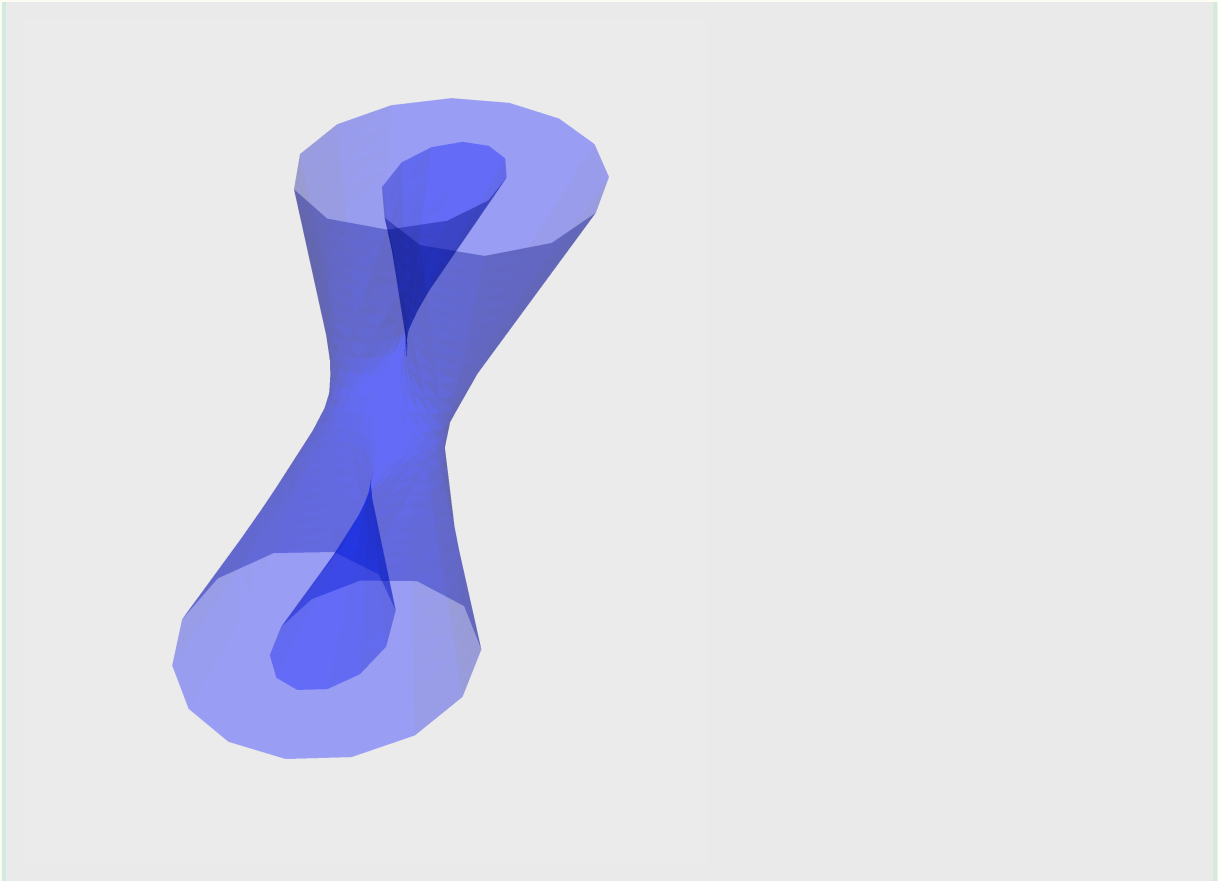$$\text{PPoint}\left[\text{Cos}[\#1] + \frac{6.55301\times10^7\ \text{Cos}[\#1]\ \text{Sin}[\#1]\ \#2}{\sqrt{2\,982\,076\,586\,042\,449\ \text{Cos}[\#1]^2 + 5\,964\,153\,172\,084\,900\ \text{Sin}[\#1]^2}}\ \&,\right.$$

$$\frac{54\,608\,393\ \text{Sin}[\#1]}{77\,227\,930} + \frac{1.2\ \text{Sin}[\#1]^2\ \#2}{\sqrt{\frac{2\,982\,076\,586\,042\,449\ \text{Cos}[\#1]^2}{5\,964\,153\,172\,084\,900} + \text{Sin}[\#1]^2}}\ \&,$$

$$\left.\#2\ \&,\ \text{PRange} \rightarrow \{\{-\pi, \pi\}, \{-\pi, \pi\}, \{25, 25\}\}\right]$$

Walls and pipes
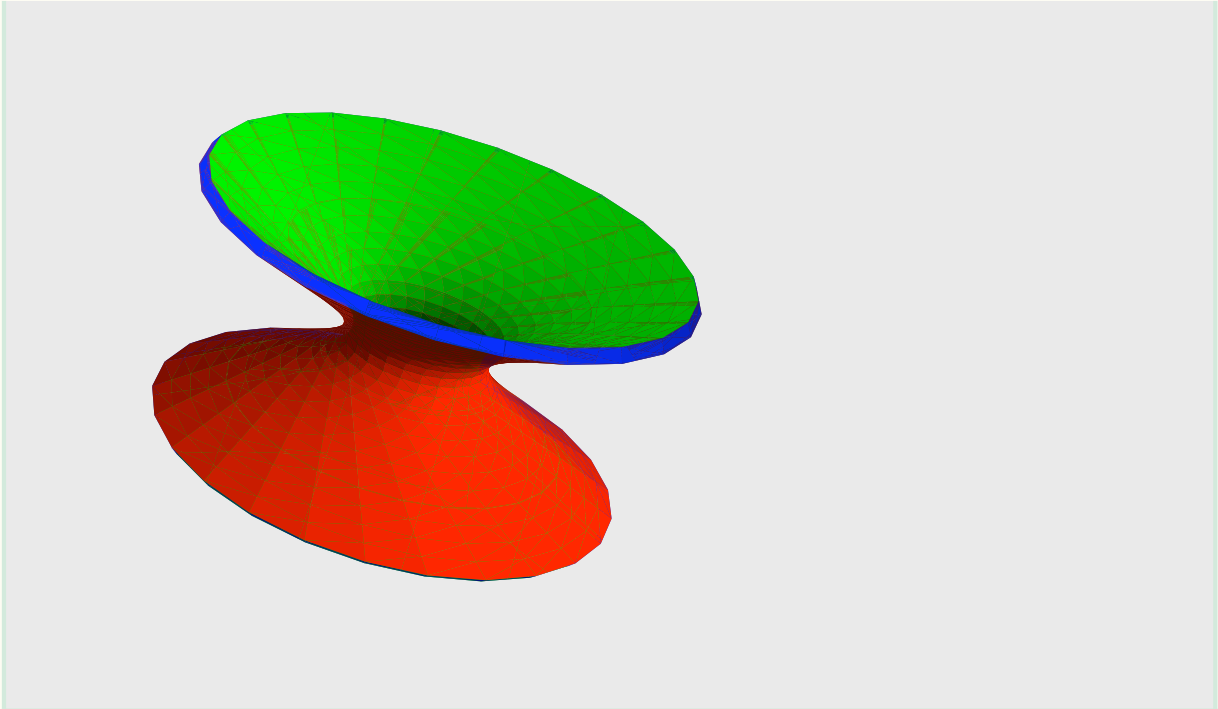
In real applications, objects have a finite thickness. An example of wall has been given for a polygon, it can be extended to any object. Here a thick hyperboloid of one sheet is drawn. One can see the three parts of the wall: the inner face in green, the outer face in red and the edge in blue.

```
h = HyperboloidOfOneSheet[];
w = Wall[h, .2];
Draw3D[Opacity[1], Paint[w, {Red, Green, Blue}]]
```
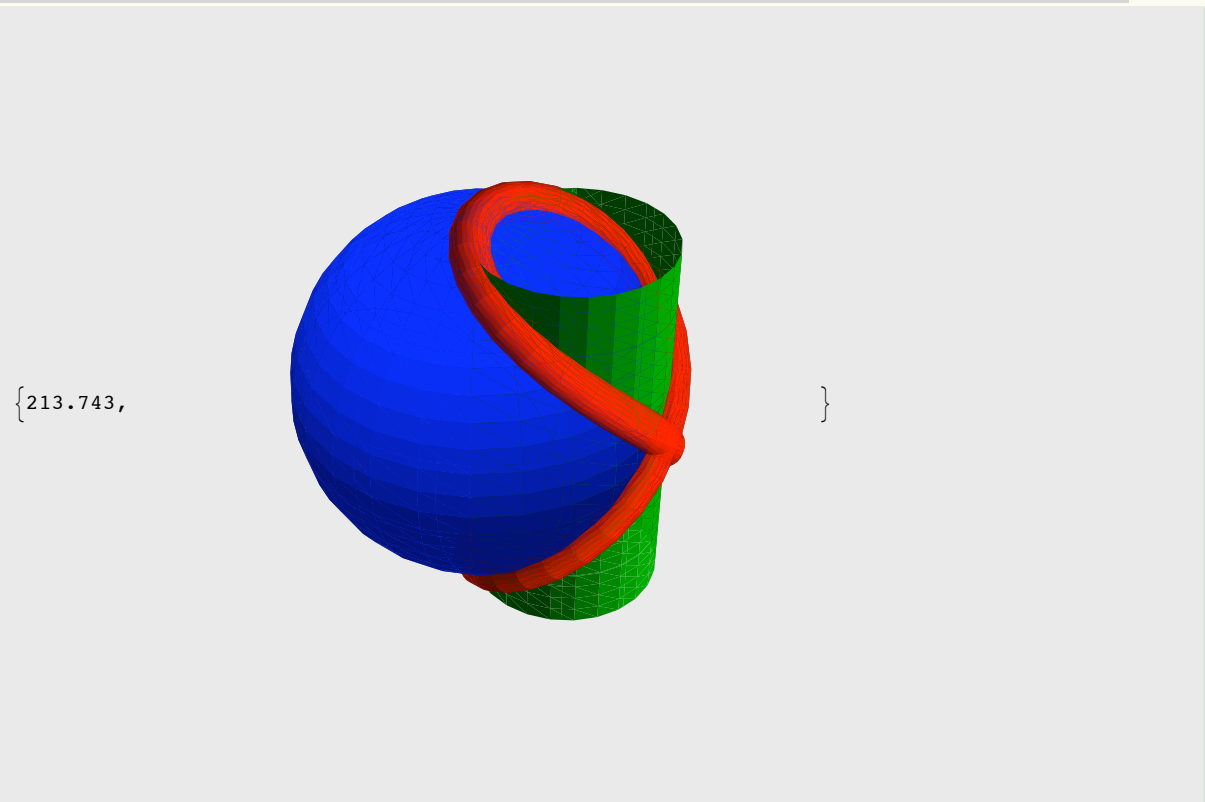
A pipe is a surface whose cross section is a circle. The example is given for a Viviani's window, the intersection of a sphere with a revolution cylinder tangent to the sphere and passing through its center. The cylinder is returned from `ECylinder` as a cartesian quadric *ccy*. To adapt its shape to the sphere, its parametric representation is obtained by applying `QuadricElements` to *ccy* and extracting the parametric representation given by the rule `QuadricPoint`. Last the parameter range `PRange` is updated. A pipe, like a wall, is a complicated surface. Although its evaluation is rather fast (1.64 s), its display takes a noticeable time (190 s) for a definition of 25 points per isocurve.

Tour of Geometrica

```
s = ESphere[];
ccy = ECylinder[CPoint[0, 0, 1], ECircle[CPoint[1 / 2, 0], 1 / 2]];
mcy = QuadricPoint /. (QuadricElements @@ ccy);
cy = mcy /. _Rule → (PRange → {{0, 2 Pi}, {-1, 1}, {25, 25}});
v = VivianiWindow[];
p = Pipe[v, .1];
Timing@Draw3D[Opacity[1], Paint[{s, cy, p}, {Blue, Green, Red}]]
```

Tour of Geometrica

$$\left\{213.743, \right\}$$

Tour of Geometrica

The user can keep playing with pipes by making the radius vary.

```
s = ESphere[];
ccy = ECylinder[CPoint[0, 0, 1], ECircle[CPoint[1 / 2, 0], 1 / 2]];
mcy = QuadricPoint /. (QuadricElements @@ ccy);
cy = mcy /. _Rule → (PRange → {{0, 2 Pi}, {-1, 1}, {25, 25}});
v = VivianiWindow[];
```