

Geometrica02

Tour of Geometrica

Calling Geometrica

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

```
<< Geometrica`Geometrica02`
```

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

Geometrica and Mathematica

list processing
symbolic facilities
functional programming
<i>Draw</i> and <i>Draw3D</i> 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 theory of curves)

Solving an Elementary Problem

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

Point, Line, and First Drawing

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

The straight line is the y-axis.

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

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

```
g1 = Draw[Red, m, d];
```



Bound Point and Euclidean Line

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

```
m1 = Pointer[d, t]
r = t -> 1;
g2 = Draw[Blue, m1 /. r];
d1 = ELine[m, m1]
```

```
g3 = Draw[Blue, d1 /. r];
```

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

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



Geometrical Function

The other line d_2 is perpendicular in m_1 to d_1 .

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

```
g4 = Draw[Blue, d2 /. r];
```

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



Derivative of a Geometrical Function

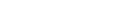
When the parameter t varies, the line d_2 moves. Between two infinitesimally close positions, the two at a point of the envelope.

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

```
g5 = Draw[Blue, d21 /. r];
```

```
m_p = d2 ∩ d21
```

```
g6 = Draw[Red, m_p /. r];
```



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

```
CPoint[t2, 2t]
```

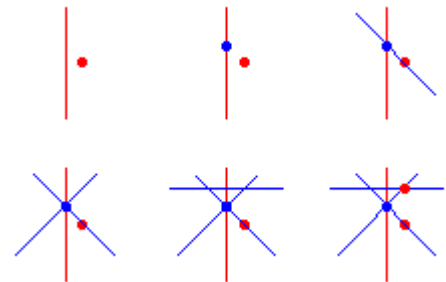
Animation and StoryBoard

One can now recapitulate the construction by using the command `Movie`, which produces the six figures integrating the $n+1$ th figure to the n th one.

```
Movie[g1, g2, g3, g4, g5, g6]
```

The movie is adapted to a demo but, to keep the trace of the construction, it is more appropriate to use `Storyboard` which puts the figures in a table.

```
Storyboard[{g1, g2, g3, g4, g5, g6}, 3];
```



Viewing the Envelope Process

We are now curious to see the motion of the line d_2 . We then make a table of the graph of the line for a range of t . By selecting all of the graphics cells and pressing "Animate selected cells" in the Cell menu, the line will appear as the envelope of the lines.

```
g7 = Table[Draw[Red, LineOrigin[m], d2, DisplayFunction -> Identity],  
           {t, -2, 2, .2}];  
g8 = Movie[g7];
```

Study of a Conic

To find the equation of the envelope, we eliminate t between the coordinates of m_p .

```
l1 = {x, y};  
l2 = List @@ mp;  
eqs = Thread[Equal[l1, l2]];  
eq = Eliminate[eqs, t]
```

```
y2 == 4x
```

Geometrica can recognize the nature of the conic from its Cartesian equation $rx^2 + 2sxy + ty^2 + 2ux + 2vy + w = 0$.

```
p = CConic[0, 0, 1, -2, 0, 0]  
Parabola[0, 0, 1, -2, 0, 0]
```

It can also determine its focus and directrix.

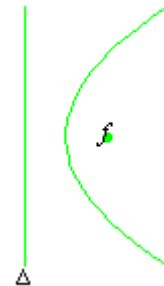
```
{f, A} = {Focus[p], Directrix[p]}
{CPoint[1, 0], CLine[-4, 0, -4]}
```

```
delta = Simplify[delta]
```

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

Summary

```
g9 = Draw[Thickness[.01], Green, p, f, A,
          Black, Legend[{"f", "A"}, {f, A}]];
```



```
Show[Last[g8], g9];
```

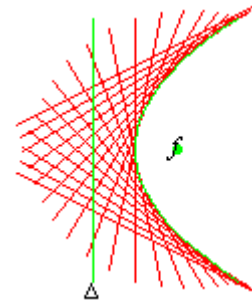


Illustration in 3D

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

3D Conic

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

```
p3D = To3D[p]
CConic[Parabola[0, 0, 1, -2, 0, 0],
        CPlane[0, 0, 1, 0]]
```

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

```
m3D = Pointer[p3D, n]
CPoint[ $\frac{n^2}{4}$ , n, 0]
```

The tangents are obtained using `ELine`.

```
tg = ELine[m3D, p3D]
```

$$\text{CLine}\left[\text{CPlane}[0, 0, 1, 0], \text{CPlane}\left[-2, n, 0, -\frac{n^2}{2}\right]\right]$$

Cylinder

A cylinder is defined by a curve, here the parabola, and a generator parallel to a fixed direction Oa which has been chosen as the point of Oz of height 6.

$$\mathbf{cy} = \text{Cylinder}[\text{CPoint}[0, 0, 6], \mathbf{p3D}]$$

$$\text{PPoint}\left[\frac{\#1^2}{4} \&, \#1 \&, 6 \#2 \&, \text{FRange} \rightarrow \{(-\pi, \pi), \{0, 1\}, \{25, 2\}\}\right]$$

The planes tangent to the parabolic cylinder are represented by parallelograms of vertices: the contact point with the parabola, the point of parameter 0 of the tangents tg to the parabola, the point of height 6 on the vertical line passing through the contact point,

$$\mathbf{a} = \text{Pointer}[tg, 0]$$

$$\text{CPoint}\left[-\frac{n^2}{4+n^2}, \frac{n^3}{8+2n^2}, 0\right]$$

and the point of height 6 on the vertical line passing through the contact point.

$$\mathbf{b} = \text{Translate}[\mathbf{m3D}, \text{CPoint}[0, 0, 6]]$$

$$\text{CPoint}\left[\frac{n^2}{4}, n, 6\right]$$

The parallelogram is then given by:

$$\mathbf{pp} = \text{Parallelogram}[\mathbf{m3D}, \mathbf{a}, \mathbf{b}]$$

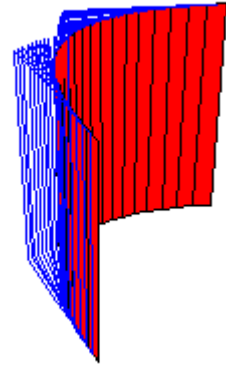
$$\begin{aligned} \text{Segment}\left[\text{CPoint}\left[\frac{n^2}{4}, n, 0\right], \right. \\ \text{CPoint}\left[-\frac{n^2}{4+n^2}, \frac{n^3}{8+2n^2}, 0\right], \\ \text{CPoint}\left[-\frac{n^2}{4+n^2}, \frac{n^3}{8+2n^2}, 6\right], \\ \left. \text{CPoint}\left[\frac{n^2}{4}, n, 6\right], \right. \\ \left. \text{CPoint}\left[\frac{n^2}{4}, n, 0\right]\right] \end{aligned}$$

The full figure can now be drawn.

```

ta = Table[pp, {n, -3, 3, .2}];
Draw3D[Paint[cy, Red], Blue, ta];

```



CAD Functions

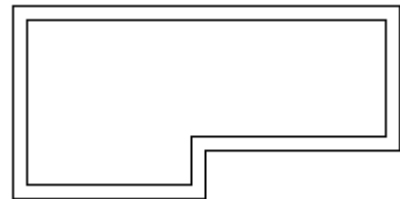
Walls as Examples of Parallel Polygons

A line L' is parallel to a line L when it is generated by the end of a segment attached and normal to L assumes that L represents a continuous and derivable function, which is not true for a polygon that has a vertex. A convention is introduced in *Geometrica* to define a parallel polygon. It is illustrated for 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];
s_p = Parallel[s, .3];
Draw[s, s_p];

```



The wall has been defined with openings that can be seen if the option `Ribbon` is used. The inner poly "ribbon" can be painted to distinguish the full and empty parts of the wall.

```

s_p = Parallel[s, .3, Ribbon -> True];
col = {Black, White, Black, Black,
       Black, White, Black, Black,
       White, Black, Black, Black,
       White, Black};
Draw[Paint[s_p, col]];

```



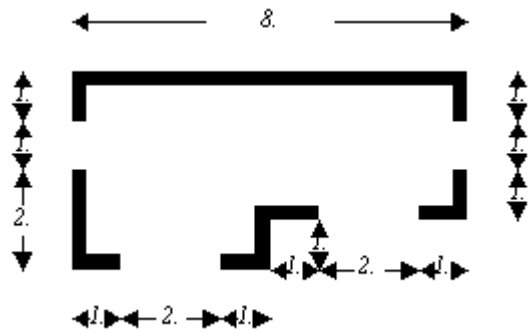
Putting Dimensions on a Technical Drawing

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

```

dim = Dimension[s];
Draw[Paint[s_p, col],
      dim];

```



Optics Functions

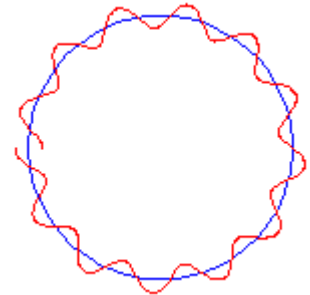
Paraxial lines are generated in the same way as parallel lines, but the distance between the lines var to a prescribed function.

```
{m1, m2} = CPoint[{0, 1}, 0];
c = ECircle[m1, m2];
l = Paraxial[c, .1 Sin[2. 2 π #] &]
```

```
PPoint[Cos[#1] - 0.1 Cos[#1] Sin[12.5664 #1] &,
Sin[#1] - 0.1 Sin[#1] Sin[12.5664 #1] &,
PRange -> {-π, π, 25}]
```

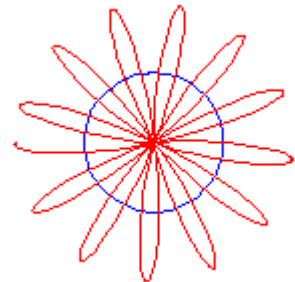
```
Draw[Blue, c, Red, l];
```

```
Ellipse[1, 0, 1, 0, 0, -1]
```

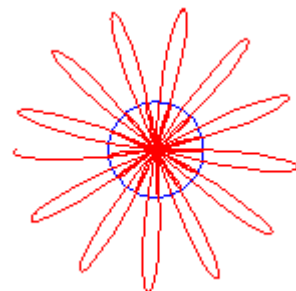


The shape of the paraxial line may take fancier shapes for larger amplitudes of the oscillation.

```
l = Paraxial[c, Sin[2. 2 π #] &];
Draw[Blue, c, Red, l, PlotRange -> All];
```



```
l = Paraxial[c, 2 Sin[2. 2 π #] &];
Draw[Blue, c, Red, l, PlotRange -> All];
```



The functions `Parallel` and `Paraxial` apply to any curve or surface. This is the paraxial curve of a hel

```
h = PPoint[2 Cos[#] &,
          2 Sin[#] &,
          # &,
          PRange -> {0, 4 Pi, 50}];
l = Paraxial[h, 2 Sin[8. 2 π #] &]
```

```
PPoint[2 Cos[#1] - 2 Cos[#1] Sin[50.2655 #1] &,
        2 Sin[#1] - 2 Sin[#1] Sin[50.2655 #1] &,
        #1 &,
        PRange -> {0, 4 π, 50}]
```

```
Draw3D[Blue, h, Red, l];
```

