Programming an Anamorphic Surface
Abstract

The Programmable Interface in VirtualLab Fusion enables the definition of customized freeform surfaces. This example shows how to define an anamorphic surface as a height function, and especially with the surface gradient analytically given as well. Surface specification parameters, like conical constant, curvature, and polynomial coefficients values, are all customizable for the user.
The height function of an anamorphic surface is defined by

\[ h(x, y) = \frac{C_x x^2 + C_y y^2}{1 + \sqrt{1 - (1 + k_x)C_x^2 x^2 - (1 - k_y)C_y^2 y^2}} + \sum_{i=2}^{1} a_{r,i} \left[ (1 - a_{p,i})x^2 + (1 + a_{p,i})y^2 \right]^i \]

\( C_x \) and \( C_y \) are the curvatures in \( x \) and \( y \) direction with

\[ C_x = \frac{1}{R_x} \]
\[ C_y = \frac{1}{R_y} \]

\( R_x \) and \( R_y \) are the radii of the surface and \( k_x \) and \( k_y \) the conical constants in \( x \) and \( y \) direction. \( a_{r,i} \) and \( a_{p,i} \) are polynomial coefficients of the surface.

Task:
Use Programmable Interface to generate an anamorphic interface according to the equation.
Programming a Anamorphic Surface (Height)

Global Parameters (User Defined)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Allowed range</th>
</tr>
</thead>
<tbody>
<tr>
<td>double ConicalConstantX</td>
<td>0</td>
<td>-1000 - 1000</td>
</tr>
<tr>
<td>double ConicalConstantY</td>
<td>0</td>
<td>-1000 - 1000</td>
</tr>
<tr>
<td>double CurvatureX</td>
<td>348.7992127</td>
<td>0 - 1000</td>
</tr>
<tr>
<td>double CurvatureY</td>
<td>0</td>
<td>0 - 1000</td>
</tr>
<tr>
<td>double PolynomialCoefficients</td>
<td>{-2948903.66, 73895084200, -3.46, 8.19}, {0.188,-1.35,-0.767,-1.02}</td>
<td>0 - 1000</td>
</tr>
</tbody>
</table>

```java
double height = 0.0;

// help variables
double curvX = CurvatureX;
double curvY = CurvatureY;
int polynomialDegree = PolynomialCoefficients.GetLength(0);

if (there are not two rows in the coefficients table -> this is an error)
if (PolynomialCoefficients.GetLength(1) != 2) {
    return 0;
}

// calculate polynomial term for position (x,y)
double polynomialTerm = 0;
for (int Index = 0; Index < polynomialDegree; index++) {
    double a_r1 = PolynomialCoefficients[index, 0];
    double a_p1 = PolynomialCoefficients[index, 1];
    polynomialTerm += a_r1 * Math.Pow((1 - a_p1) * x * x + (1 + a_p1) * y * y, index + 2);
}

// calculate height for position (x,y)
height = (curvX * x * x + curvY * y * y) / 
       (1 + Math.Sqrt((1 - (1 + ConicalConstantX) * curvX * curvX + x * x - (1 + ConicalConstantY) * curvY * curvY * y * y)) + polynomialTerm);

return height;
```
Programming a Anamorphic Surface (Gradient)

```java
Vector2D gradient = new Vector2D();
gradient.X = 0;
gradient.Y = 0;

//help variables
double v = Math.Sqrt(1 - (1 + ConicalConstantX) * CurvatureX * x * x - (1 + ConicalConstantY) * CurvatureY * y * y);
double sumTermX = 0, sumTermY = 0;

int polynomialDegree = PolynomialCoefficients.GetLength(0);
for (int index = 0; index < polynomialDegree; index++) {
    double a_i = PolynomialCoefficients[index, 0];
    double a_pi = PolynomialCoefficients[index, 1];
    sumTermX += 2 * (index + 2) * (1 - a_pi) * x * a_i * Math.Pow((1 - a_pi) * x * x + (1 + a_pi) * y * y, index + 1);
    sumTermY += 2 * (index + 2) * (1 + a_pi) * y * a_i * Math.Pow((1 - a_pi) * x * x + (1 + a_pi) * y * y, index + 1);
}

gradient.X = 2 * CurvatureX * x / (1 + v) - (CurvatureX * x * x + CurvatureY * y * y) *
            (-2 * CurvatureX * CurvatureX * x * (1 + ConicalConstantX)) / (2 * v * (1 + v) * (1 + v)) +
            sumTermX;

gradient.Y = 2 * CurvatureY * y / (1 + v) - (CurvatureX * x * x + CurvatureY * y * y) *
            (-2 * CurvatureY * CurvatureY * y * (1 + ConicalConstantY)) / (2 * v * (1 + v) * (1 + v)) +
            sumTermY;

return gradient;
```
<table>
<thead>
<tr>
<th>title</th>
<th>Programming an Anamorphic Surface</th>
</tr>
</thead>
<tbody>
<tr>
<td>document code</td>
<td>CZT.0034</td>
</tr>
<tr>
<td>version</td>
<td>1.0</td>
</tr>
<tr>
<td>toolbox(es)</td>
<td>Starter Toolbox</td>
</tr>
<tr>
<td>VL version used for</td>
<td>7.4.0.49</td>
</tr>
<tr>
<td>simulations</td>
<td></td>
</tr>
<tr>
<td>category</td>
<td>Feature Use Case</td>
</tr>
<tr>
<td>further reading</td>
<td>- How to Work with the Programmable Interface &amp; Example (Spherical Surface)</td>
</tr>
<tr>
<td></td>
<td>- Programming a Sinusoidal Surface</td>
</tr>
<tr>
<td></td>
<td>- Construction of a Truncated Pyramid Surface</td>
</tr>
</tbody>
</table>