GeniE V6.4

GenieRhinoConverter
Plug-in for Rhino 4.0 32bit and Rhino 5.0 32bit/64bit

Introduction

The plug-in GenieRhinoConverter (the updated version of Rhino4ToGenie in Genie V6.3) is a utility for exchanging curves in NURBS form, between Rhino 4.0 (32bit) or Rhino 5.0 (32bit and 64bit) and Genie V6.4. The curves from Rhino are stored in a java-script command file, which can be imported into Genie V6.3 and above, using the "File/Read Command File". The curves from Genie are stored in a *.grc-file (grc is an acronym which stands for Genie-Rhino Conversion) which can be imported into Rhino.

The plug-ins are shipped with Genie and they can be found under:

Training/Rhino_Plugin/InputFiles/32bit/GenieRhinoConverter32.rhp

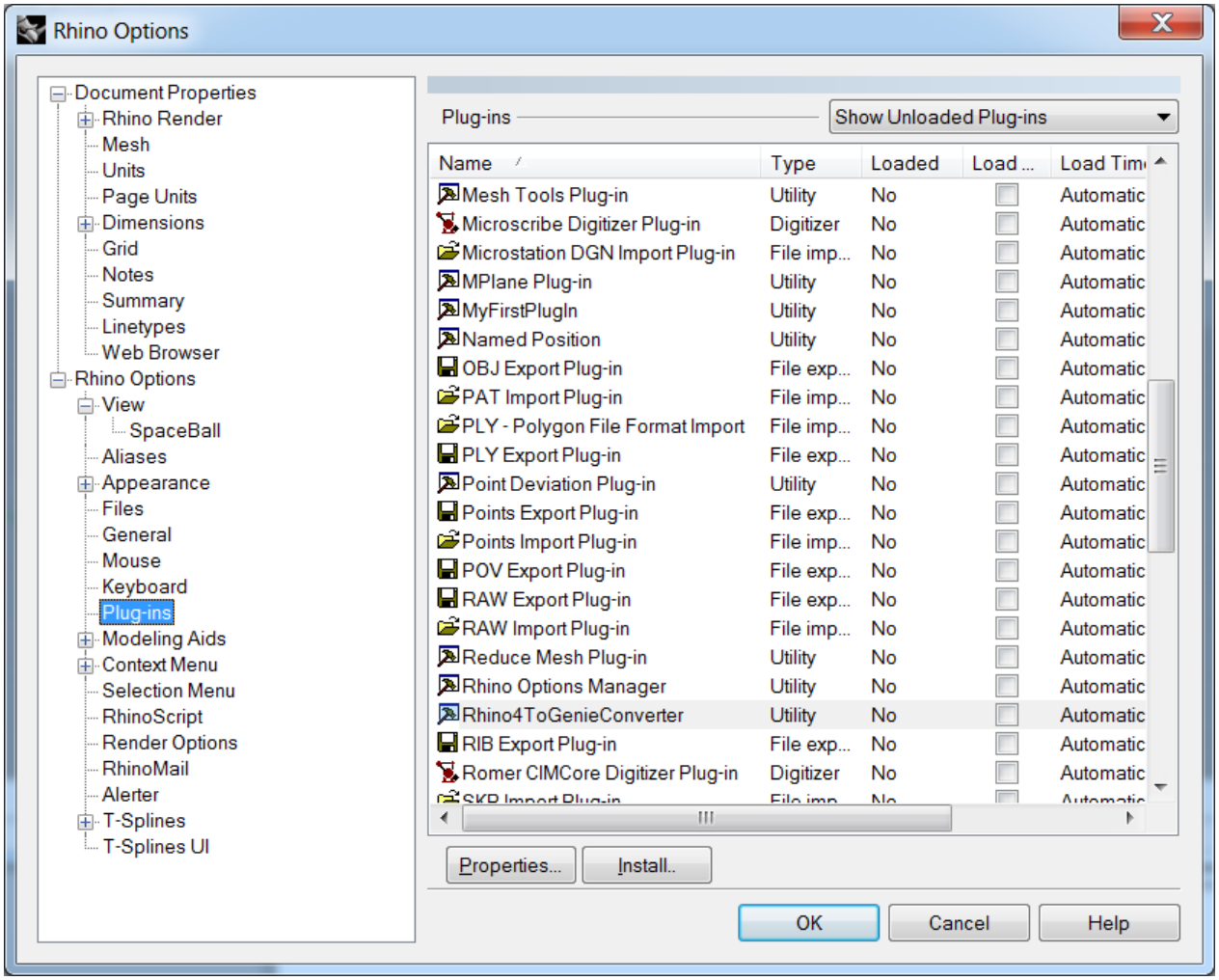
(Rhino 4.0 and Rhino 5.0 32bit)

Training/Rhino_Plugin/InputFiles/64bit/GenieRhinoConverter64.rhp

(Rhino 5.0 64bit)

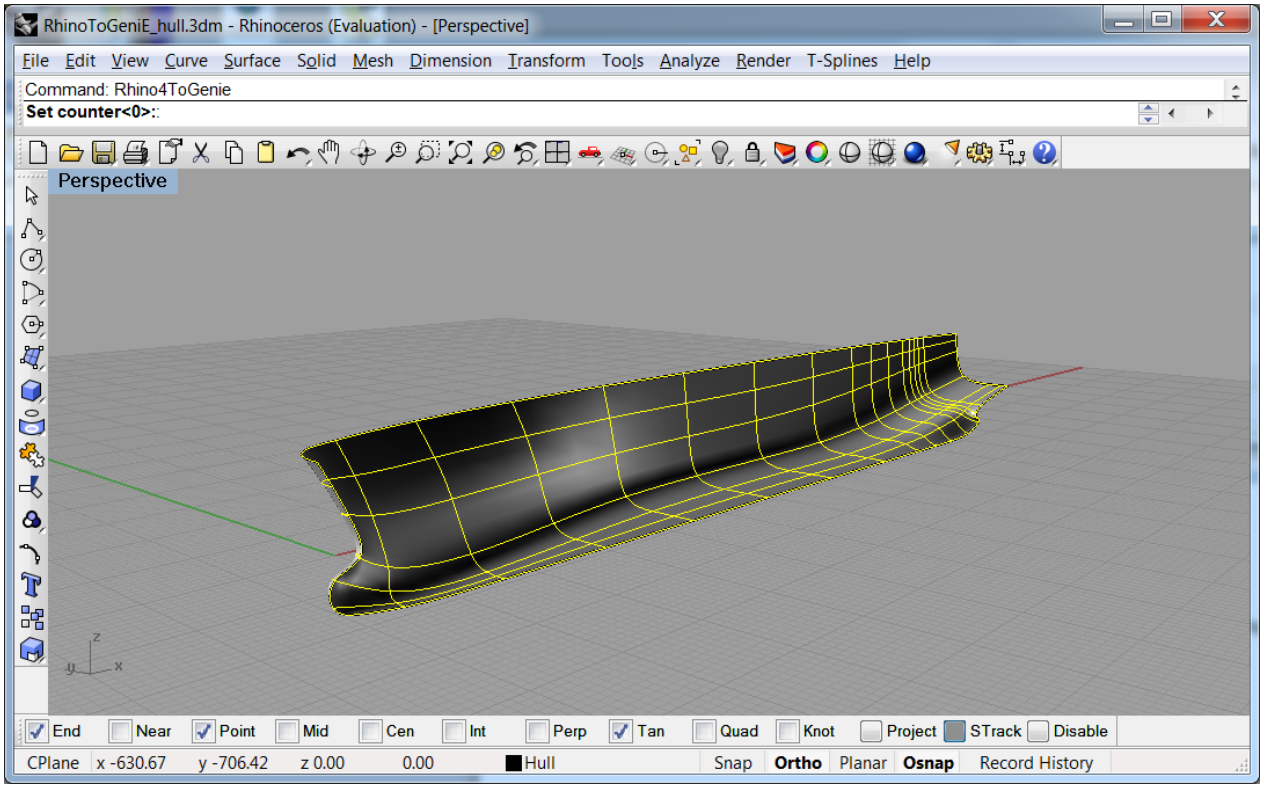
Installation

Open Rhino and select “Tools\Options”. The window of the following image appears. Select the item “Plug-ins” on the left-hand-side tree and you see the plug-ins installed. Press the button “Install” and choose the “GenieRhinoConverter32.rhp” or “GenieRhinoConverter64.rhp” file from the above location, according to the Rhino version you have.



Exporting curves from Rhino and Importing them to Genie

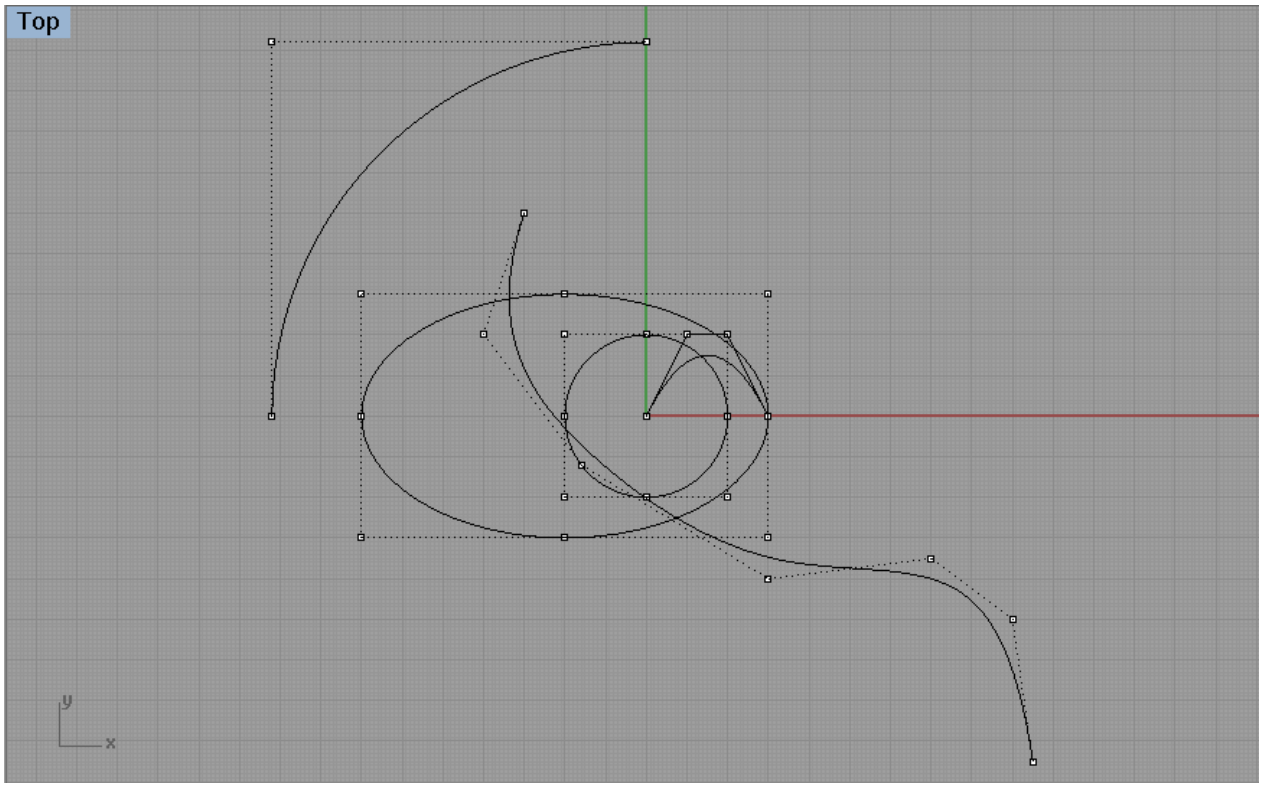
The plug-in installs one Rhino command, which can be used only from the Rhino-command-line (there is no new entry in the menus or toolbars). The user writes the command “RhinoToGenie”. Then Rhino asks for one or more curves to select or takes the already selected curves (see the next figure). After this, Rhino asks for the curve counter. Note that the java-script file, which is to be imported into Genie, uses the convention: “Curve# =…” where “#” stands for an increasing number, which starts from the curve counter. The default value is 0. Negative values are ignored.



Finally, Rhino opens the standard windows “Save As...” dialog for saving the java-script file. The file contains one-by-one the js-commands of all the selected curves and it is ready to be read by Genie.

Example

The plug-in is shipped along with a test-example, namely “simple_test_case.3dm”. A view of the example is shown in the following figure captured in Rhino 4.0.



We run the command “RhinoToGenie” and we set the counter equal to “10”. The java-script commands, stored in the output file, are the following:

```
Curve10=CreateBSpline(
          Array(Point(-30.000000,50.000000,0.000000),
               Point(-40.000000,20.000000,0.000000),
               Point(-16.000000,-12.000000,0.000000),
               Point(30.000000,-40.000000,0.000000),
               Point(70.000000,-35.000000,0.000000),
               Point(90.000000,-50.000000,0.000000),
               Point(95.000000,-85.000000,0.000000)),
          Array(0.000000,0.000000,0.000000,
               1.000000,2.000000,3.000000,
               4.000000,4.000000,4.000000),
          3);
```

```
Curve11=CreateNURBS(
          Array(Point(30.000000,0.000000,0.000000),
               Point(30.000000,30.000000,0.000000),
               Point(-20.000000,30.000000,0.000000),
               Point(-70.000000,30.000000,0.000000),
               Point(-70.000000,0.000000,0.000000),
               Point(-70.000000,-30.000000,0.000000),
               Point(-20.000000,-30.000000,0.000000),
               Point(30.000000,-30.000000,0.000000),
               Point(30.000000,0.000000,0.000000)),
          Array(0.000000,0.000000,1.570796,1.570796,
               3.141593,3.141593,4.712389,4.712389,
               6.283185,6.283185),
          Array(1.000000,0.707107,1.000000,0.707107,1.000000,
               0.707107,1.000000,0.707107,1.000000),
          2);

Curve12=CreateBSpline(
          Array(Point(0.000000,0.000000,0.000000),
               Point(10.000000,20.000000,0.000000),
               Point(20.000000,20.000000,0.000000),
               Point(30.000000,0.000000,0.000000)),
          Array(0.000000,0.000000,0.000000,
               1.000000,1.000000,1.000000),
          3);

Curve13=CreateBSpline(
          Array(Point(0.000000,0.000000,0.000000),
               Point(10.000000,20.000000,0.000000),
               Point(20.000000,20.000000,0.000000),
               Point(30.000000,0.000000,0.000000)),
          Array(0.000000,22.360680,32.360680,54.721360),
          1);
```

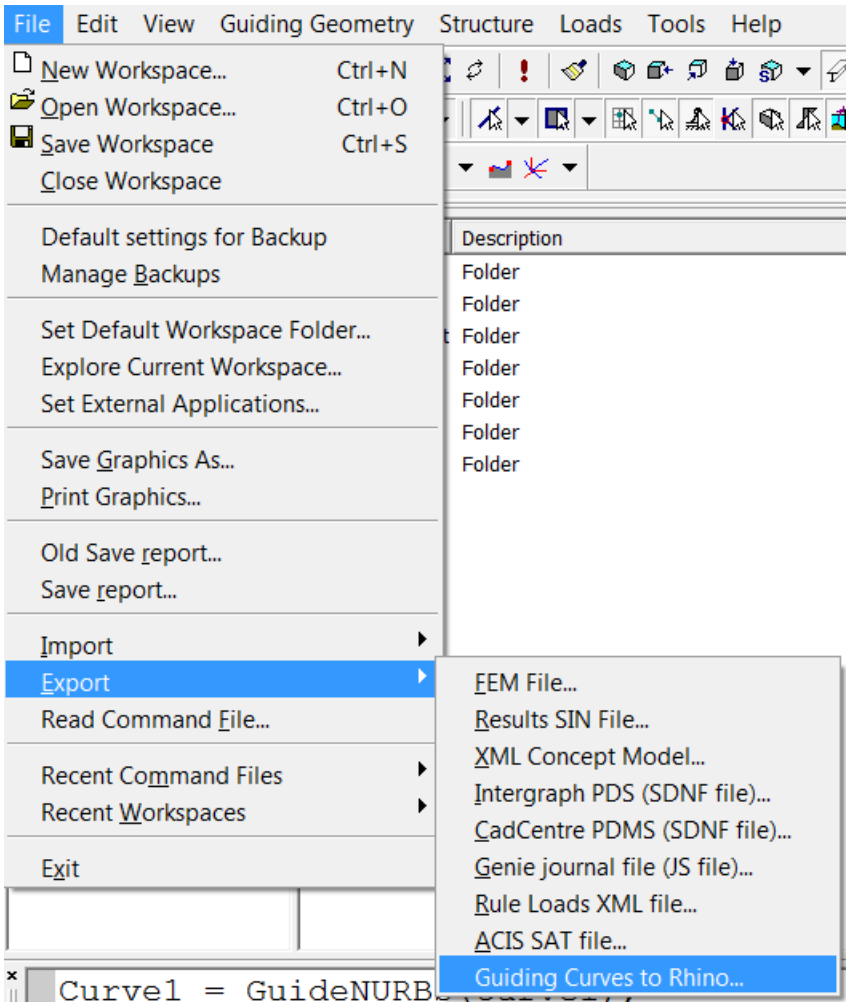
```
Curve14=CreateNURBS(
          Array(Point(20.000000,0.000000,0.000000),
               Point(20.000000,20.000000,0.000000),
               Point(0.000000,20.000000,0.000000),
               Point(-20.000000,20.000000,0.000000),
               Point(-20.000000,0.000000,0.000000),
               Point(-20.000000,-20.000000,0.000000),
               Point(-0.000000,-20.000000,0.000000),
               Point(20.000000,-20.000000,0.000000),
               Point(20.000000,0.000000,0.000000)),
          Array(0.000000,0.000000,31.415927,31.415927,
               62.831853,62.831853,94.247780,94.247780,
               125.663706,125.663706),
          Array(1.000000,0.707107,1.000000,0.707107,1.000000,
               0.707107,1.000000,0.707107,1.000000),
          2);

Curve15=CreateNURBS(
          Array(Point(-92.000000,0.000000,0.000000),
               Point(-92.000000,92.000000,0.000000),
               Point(-0.000000,92.000000,0.000000)),
          Array(0.000000,0.000000,144.513262,144.513262),
          Array(1.000000,0.707107,1.000000),
          2);
```

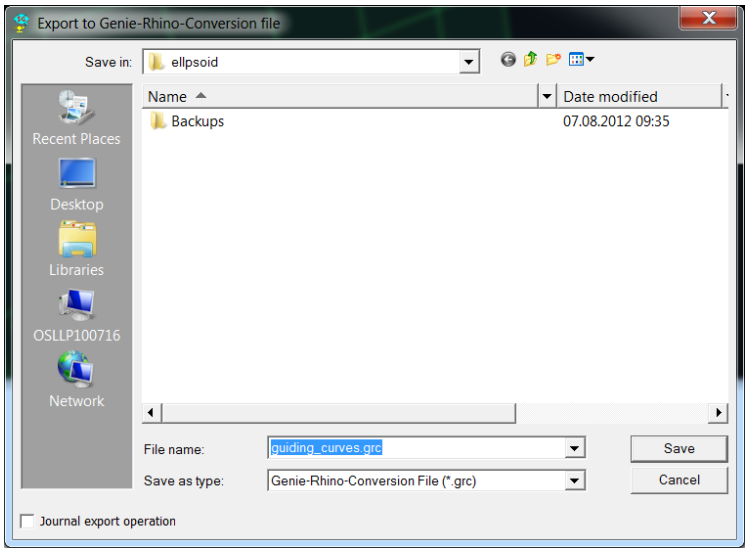
The interested user may verify the control points with the ones shown in the above figure and may distinguish between the b-splines (Curve10, Curve12 and Curve13) and the rational b-splines (Curve11, Curve14 and Curve15). He may also find that the poly-line is actually Curve13, represented as a b-spline of first degree and all the conic sections (the ellipse, the circle and the circular arc) are represented as rational b-splines of second degree.

Exporting curves from Genie and Importing them to Rhino

Genie is able to export the currently selected guiding curves. Select the menu item: File/Export/Guiding Curves to Rhino



Genie gives a prompt to save the curves in an output file of type *.grc.



The *.grc-file has the following format:

```
BSplineCurve3D {
Name[Curve3],
Layer[],
ControlPoints[Point(7.5:7.5:10) Point(5:7.5:10) Point(5:5:10)
 Point(5:2.5:10) Point(7.5:2.5:10) Point(10:2.5:10) Point(10:5:10)
Point(10:7.5:10) Point(7.5:7.5:10)],
KnotVector[0 0 1.5708 1.5708 3.14159 3.14159 4.71239, 4.71239
6.28319 6.28319],
Weights[0.707107 1 0.707107 1 0.707107 1 0.707107],
Degree[2]
};
```

After saving the *.grc-file, one can read it from Rhino by entering the command: “GenieToRhino”. Note that the curves in Genie have a name, which is preserved when importing them into Rhino. So, inversely, using the Rhino command “RhinoToGenie”, the program reads the curve name and if that name exists, then it is exported along with the curve as its Genie-name.

In order to be able to read, modify and delete the Genie-names, the plug-in offers the following three commands:

GetName, SetName and DelName

The full list of commands offered by the plug-in can be found by selecting it in the Rhino-Options dialog and on the RMB select the “Commands”. Then a dialog like the one below is shown up.