Offset, Lines, Shapes, Freehand Eraser and Contour Edition

QUICKCARD – VERSION 1.5

1. Overview

ToolsOnSurface is a suite of interactive drawing tools that works on 3D surfaces, as opposed to the native Sketchup tools that always draw within a 2D plane.

In version 1.5, the following tools are available:

- **Line**, single and chained, with angle management
- **Shapes**: rectangle, circle by radius/diameter, circle by 3 points, polygon, arc, parallelogram, ellipse and sector.
- **Contour Offset**, analog to its native cousin, but on curved surfaces.
- **Freehand / Polyline**, allowing drawing polylines, without and with inference, in point & click mode and in drag-over mode.
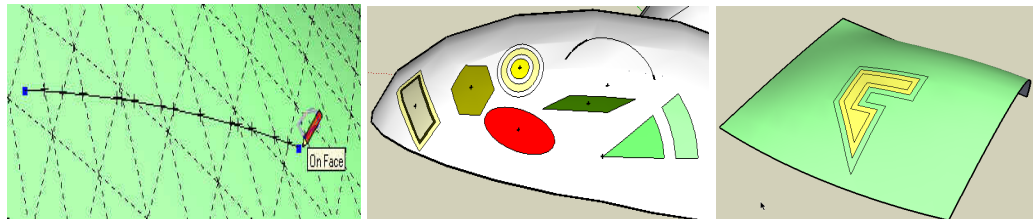
All contours generated by ToolsOnSurface can be further edited via two specific tools:

- **Eraser**, to remove individual segments
- **Editor**, to move, add and delete vertices in the contours
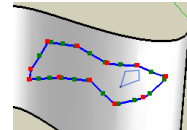
The tools also work on planar surfaces or even without a surface. In the latter case, you can specify the plane in which you draw.

All drawings can be done as plain edges or as construction lines (i.e. guides). All tools implement an inference mechanism (axes, remarkable angles, etc…) with an optional lock. There are several options available for each tool, available via the button palette, contextual menu and shortcuts. In most situations, drawing parameters (length, angle, etc…) can be entered via the VCB in any valid unit.

2. Installation and Configuration

The plugin is published as a Zip file *ToolsOnSurface 1.5.zip*.

The plugin requires **LibFredo6 v3.4 or higher**. This is a shared library used by my other scripts. LibFredo6 is delivered as a Zip file *LibFredo6 3.4.zip* (or higher version).

1) Installation of the Plugin

▪ **Unzip** *'ToolsOnSurface 1.5.zip'*, by doing a *Select All* and then *Extract* **in the Plugins directory of Sketchup**, answering **YES** to all prompts for file overwriting, in order to make sure you reconstruct the <u>folder hierarchy</u>. NEVER change the name of files or move them from their folder. The footprint should be (Windows environment):

Unzip *'LibFredo6 3.4.zip'*, in the same way.

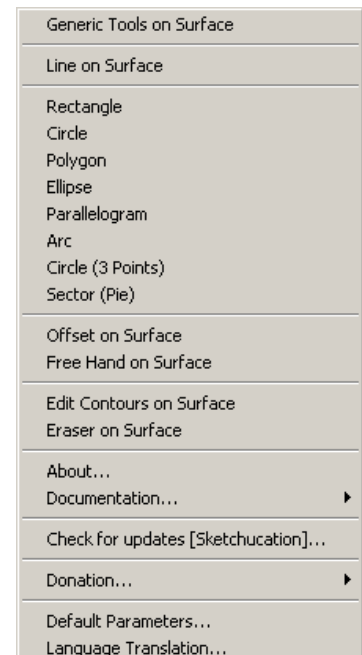The footprint should be (Windows environment):

Name	Type
⊟ c:\program files\google\google sketchup 7\Plugins\	Folder
⊞ 📁 LIBFREDO6_Dir_34	Folder
⊞ 📁 TOS_Dir_15	Folder
📄 LibFredo6.rb	RB File
📄 ZLoader__OnSurface.rb	RB File

On Mac, You may have the choice for the directory for unzipping, but **it must be the <u>same</u> as where you installed LibFredo6**:

Macintosh HD/Library/Application Support/Google SketchUp 7/SketchUp/plugins

or *Macintosh HD/<user>/Application Support/Google SketchUp 7/SketchUp/plugins*

▪ **Sketchup versions**: SU6 and SU7[1].

▪ **SU Extension**: ToolsOnSurface is installed as an extension of Sketchup. So it can be enabled and disabled via the menu *Windows > Preferences > Extensions*.

▪ **Language**: Initially *English* and *French*[2]

▪ **Menus**: by default, installed in *Tools > Fredo6 Collection > ToolsOnSurface*

▪ **Icon toolbar:** by Default *'Tools On Surface'*. 14 icons are available[3]. You can however configure the plugin so that fewer icons are shown, while keeping all commands still available in the menu. *See next section.*

▪ **Default Parameters:** configuration and specific parameters can be modified via a dialog box (*see below*).

(Menu列表:)

Generic Tools on Surface

Line on Surface

Rectangle
Circle
Polygon
Ellipse
Parallelogram
Arc
Circle (3 Points)
Sector (Pie)

Offset on Surface
Free Hand on Surface

Edit Contours on Surface
Eraser on Surface

About...
Documentation...

Check for updates [Sketchucation]...

Donation...

Default Parameters...
Language Translation...

---

[1] It also works in SU5 v5.160 or higher

[2] Kind contributors can generate the translation in other languages via the menu *'Language Translation'* and publish the text file (extension **.lang**) on Sketchucation forum.
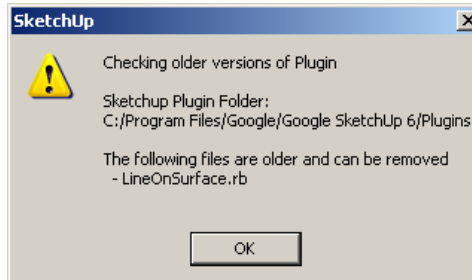
[3] Thanks and credit to **CadFather** who deigned the icons and cursors

2) Old Files and Directories

You should delete the old files and directories from previous versions:

– **Old Files**: OnSurface.rb, OffsetOnSurface.rb; LineOnSurface.rb, PolylineOnSurface.rb, EraserOnSurface.rb , PolygonOnSurface.rb

– **Old directories**: TOS_Dir, OFS_Dir

The plugin will display a Warning message about these old files the very first time you select one of the tools on surface.

```
SketchUp                                            ✕

  ⚠    Checking older versions of Plugin

       Sketchup Plugin Folder:
       C:/Program Files/Google/Google SketchUp 6/Plugins

       The following files are older and can be removed
         - LineOnSurface.rb

                    OK
```
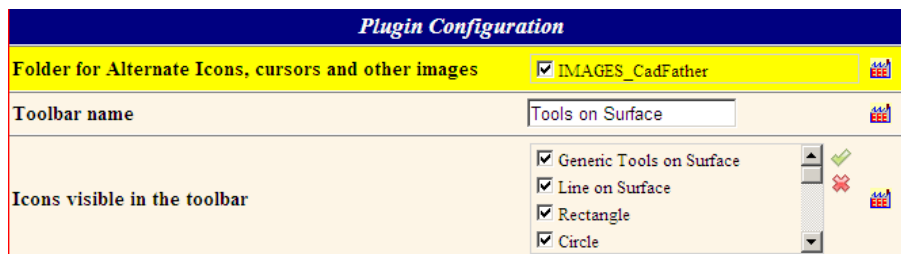
Although this is not absolutely mandatory it is recommended to avoid potential conflicts and keep your Sketchup Plugins directory clean. You can also delete the sub-folder TOS_Dir_13.

Note however that **you must NOT delete *LibTraductor.rb* from the Sketchup Plugins folder**, because it is still used by my other scripts (BezierSpline, JointPushPull, RoundEdge).

3) Configuration of the Toolbar

To help users who have real-estate problems with toolbars, ToolsOnSurface offers the possibility to configure the toolbar and choose which buttons are shown.

1) All tools are always available **from the menu**, regardless of the toolbar configuration.

2) All tools can be accessed **from within other tools**, via the lateral button palette

3) There is a **generic button**. By default it launches the Line tool, and then keeps memory of the tool you last switched to. So you are always 2 clicks away from any tool.

4) You can configure which buttons are shown in the Default Parameters box:

Plugin Configuration		
Folder for Alternate Icons, cursors and other images	☑ IMAGES_CadFather	📅
Toolbar name	Tools on Surface	📅
Icons visible in the toolbar	☑ Generic Tools on Surface ☑ Line on Surface ☑ Rectangle ☑ Circle	✓ ✗ 📅

5) You can also change the name of the toolbar, for instance if you want to group it with you own custom toolbar.

6) Finally, you can choose alternate Icons. Those designed by **CadFather** are much nicer than mine!

So a typical configuration would be to only leave the Generic button in the toolbar, and maybe one or a few buttons for the tool(s) you use frequently.

3. Common features to all Tools on Surface

1) Drawing options

You can modify the options at any time during the Drawing Operation. They are all mapped on **Toggle Function keys**, but are also available from the **contextual menu** and the **button palette**. The buttons in the palette display a small digit which corresponds to the Function key. For instance, here is the palette for the Rectangle tool:

Note: the options are preserved for the whole Sketchup session and across Sketchup sessions by default.

2) Plain Lines or Construction Lines

All tools support both modes of drawing. You can toggle between modes via the key **CTRL Alone**[4] or **F2** at any time.

In addition, you can create **Construction points at all intersections** of the created geometry (toggle **F3**). The current mode is indicated by the cursor shape, which is usually 'dashed' when drawing Construction lines.

3) Input of distances and angles in the VCB

For user convenience provides the following features for inputs entered in the VCB:

– **Decimal separator**: comma and dot are accepted, underline regardless of your local settings.

– **For Length**, numbers are interpreted in the current model units. However, **Units can be specified in the input** regardless of the current model unit. Just append the unit suffix (ex: '*3.2cm*', '*74mile*', 37.5feet, etc…).Architectural units are normally supported[5].

   Arithmetic formulas (typically with operators +, -, *, / and parentheses) are supported. For lengths, the multiplicator and divisor should be entered 'after' the length. For instance **23cm*3**, not *3*23cm*.

– **For Angles**, you have to specify the unit by appending:
   - '**d**' for degrees
   - '**r**' for radians – 1.57r is approximately 90 degrees
   - '**g**' for grades – 100g means 90 degrees
   - '**%**' for the angle tangent (i.e. slope) – 100% means 45 degree.

– **Space is the delimiter** when you have to enter several values in a row

#### 4) Inferences when drawing on Surface

The tools already had the native Sketchup inference at points, allowing stickiness at a vertex or any remarkable point (line, plane, middle point, etc).Version 1.3 and above include a mechanism to 'simulate' inferences when drawing a line between 2 points on a surface to create *stickiness* to particular directions and provide the corresponding color feedback for the line drawn (tooltip always shows which inference is used).

[4]   Except the Freehand tool, where Ctrl is used for Pausing input
[5]   As far as I understand how it works!!

This inference mechanism supports several modes:

– **If you draw in a direction which is collinear with one of the 3 axes**: color (**red**, **green**, **blue**) will indicate the corresponding axis.
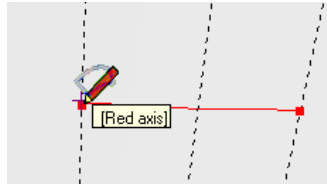

– **If you prolong the direction of the previous drawn line**: color **purple** and some simulated stickiness will tell you this is the case. Note that if the origin is at a vertex, the inference applies to all edges at this vertex. Note also that the script keeps memory about the direction of the last line drawn, and will also provide inference along this direction, even if you draw from a new origin. Inference also exists for **parallel**, **perpendicular** and **45 degree directions**.
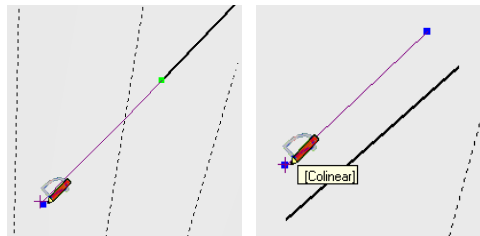
– **Inference also works for plane**, whether **axis planes** (i.e., perpendicular to an axis) or plane in which the reference line and the new lines are drawn on a face. Plane inference is indicated by a *small empty square* on the middle segment of the line drawn. For axis plane inference, the color indicates corresponds to the axis perpendicular to the pan, in <u>darker</u> color.
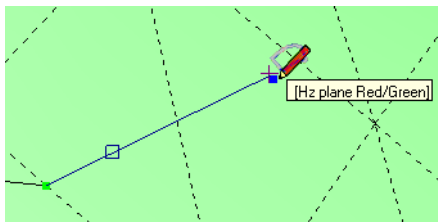

– In addition, **if you press shift while dragging the mouse**, you will **lock the direction**, whatever it is. The width of the line is **thicker** to indicate a forced direction. This is convenient, for instance, to pick a direction between two points, and then keep it preserved to draw along this direction (note that the native Sketchup Line tool only does that for the axis directions).
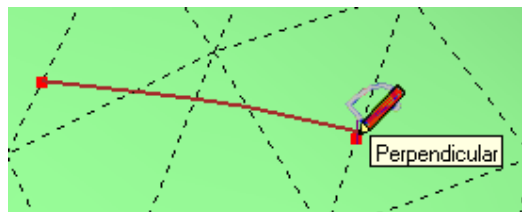
- For lines and some shapes where it applies (Sector, Circle by 3 points, parallelogram), it is possible to show a **Protractor** and observe the angle or impose it (by typing a value in degree in the VCB, followed by **'d'**). If the protractor is On, then there is some stickiness provided for angle multiple of 15 degree.
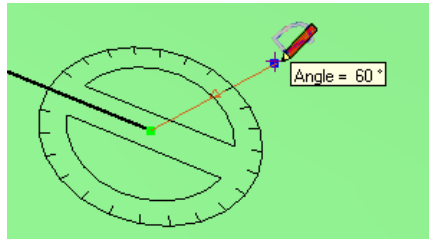
- Finally, I tried to arrange that when you have inference on, you can still terminate lines on an edge, even if the input point at the mouse is not exactly at the same location as the 'inferenced' line (this is not done in native Sketchup).
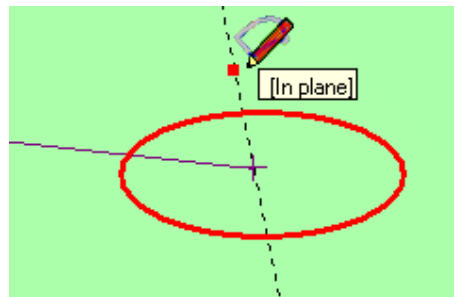
This inference mechanism is available for the Line tool, but also whenever you need to enter 2 points on a surface, as in the other tools, for instance for picking the diameter or radius of a polygon and for other shape tools. It works for Construction lines, noting however that you cannot use intersection of construction lines as true vertices.

Note: in some situation where the surface you draw upon is very close to a remarkable plane, the inference may activate for a <u>range</u> of directions, with less stickiness. I cannot do anything about that, as this is a matter of precision in Sketchup.

5) Generation in a Group

For convenience (via toggle **F6**), all generated geometry can be put into a 'TOS Group' which is <u>common</u> to all Operations on Surface and persistent during the Sketchup session. This TOS Group is generated <u>within</u> the *active* model. If you work within a group or a component, the TOS Group will be generated at this level. If you work within another group or component, another Group will be used.

By convention, the TOS Groups are named "TOS___Group". If you change this name to something different, it will force the scripts to generate a new TOS Group for the next contour drawn.

6) Redoing by double Click

All drawing tools support a Redo via Double-click. The option is also available via the contextual menu. Double Click actually allows complementing the shape with the previous parameters at any state of drawing (for instance, if you start drawing a line and double-click, then the script will keep the direction, but create the line with the previous length).

## 7) Modifying parameters AFTER operations

All tools support a change of parameters immediately AFTER the operation. For instance, once you offset a contour, you can type a new distance and get the contour adjusted. You need to avoid moving the cursor before entering the new value (or modifying the model via Undoing).

Note that due to limitations in the Sketchup Ruby API, the 'Modify After' for *Offset on Surface* can take some time if your active model contains many elements (there is a progress bar). For other tools, performance should be OK regardless of the complexity of your model.

8) Editing Contours on Surface

All contours generated by the Tools 'On Surface' can be later edited with two dedicated tools: **Eraser** and **Polyline Contour Editor**, which are described further in the present document. These tools will allow you moving, adding or erasing vertices in contours.

Note that a Contour is precisely defined by the sole fact that it has been generated by the Tools 'On Surface'. In principle, the Contour Editor will have the task to join lines drawn separately to reconstruct a full contour.
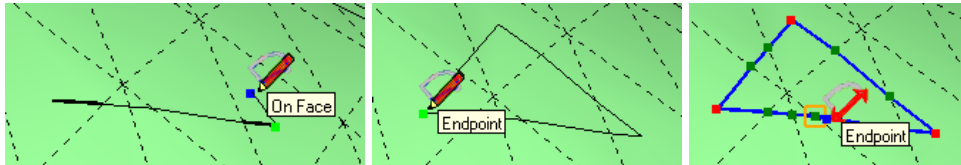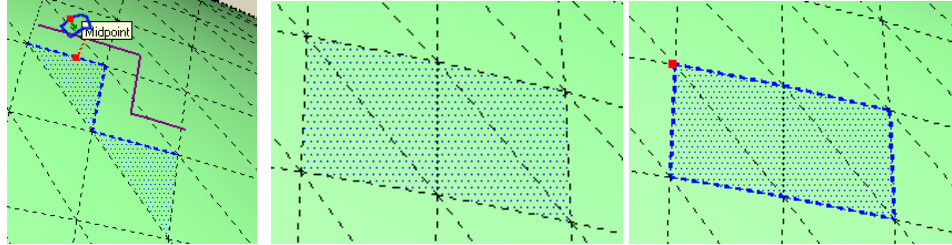
4. Offset On Surface

- **Selection of Edges and Faces**

  - **Explicit**: Select <u>first</u> Edges and Faces in the model, <u>then</u> start the tool.
    Edges must be such that they are either bordered by a single face, or if they are bordered by more, the inner face is also selected.


  - **Implicit**: Start the tool with NO active selection, and select surfaces by Mouseover (keep **Shift pressed** to select adjacent surfaces)
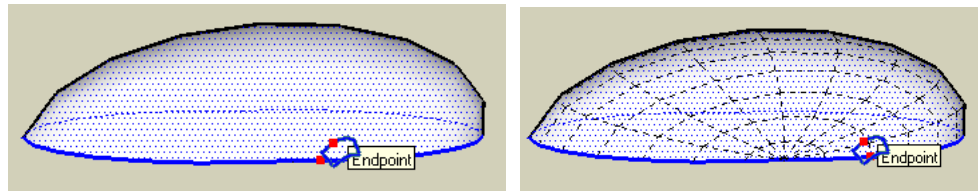
  Also, the **Toggle F5** key allows switching between selection modes: **Outer**, **Inner** or **All** contours, in case your surface has holes or embedded surfaces.

- **Entering Distance in VCB:** type number at any time, as a Length (in current model units) → ex: *"43"*. This will finish the Offset Operation.

  - **When Red Point (cross) not moved yet:** *Positive → Offset outside, Negative → Offset Inside*

  - **When Red Point (cross) already moved:** *Positive → same direction, Negative → Reverse direction*

  Inside: Distance < 0   Outside: Distance > 0
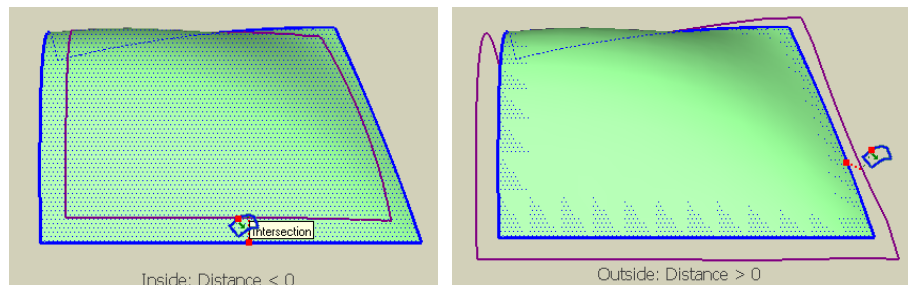
- **Dragging Contour**: **Click, Drag, Release**, or **Click, Release, Drag, and Click again**, to offset the contour, while getting visual feedback.

- **Pressing Escape** will allow getting back in Implicit selection mode.

- **Repeating Offset with last distance**: when the Red point (or cross) are 'mobile', **Double Click** to do an offset with last entered distance.
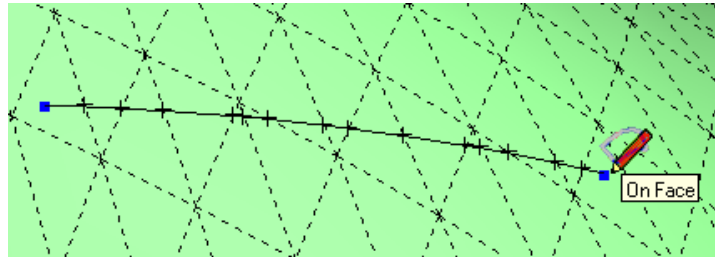
▪ **Options**

`OFFSET` `Offset = 0.0cm` `Contour Sel.`

– **Plain Lines versus Construction Lines**: **F2** or **Ctrl Alone**

– **Mark Vertices with Construction Points**: **F3**

– **Simplify Generated Contour**: **F4**

– **Select Contour type (Inner, Outer, both)**: **F5** – This only applies for implicit selection

– **Generate as Group**: **F6**

– **Treat surface as Standalone**: **F7**

– **Generate Surfaces for Outside Offset**: **F8** (Plain Line only)

– **Generate Contour as Curves**: **F9** (Plain Line only)

5. Line On Surface

This tool tries to mimic the behavior of the native Sketchup Line tool, but on Surface. It will therefore create the segments and intersections when crossing face boundaries.

This tool keeps track of the previous line drawn, so that you can use Inference relative to the previous direction (parallel, perpendicular, angle, etc…), whether or not you joint each portion of the contour.. Optionally, you can type the value of an angle in the VCB (in degree, followed by **'d'**) and show the protractor to force a particular direction on the surface.
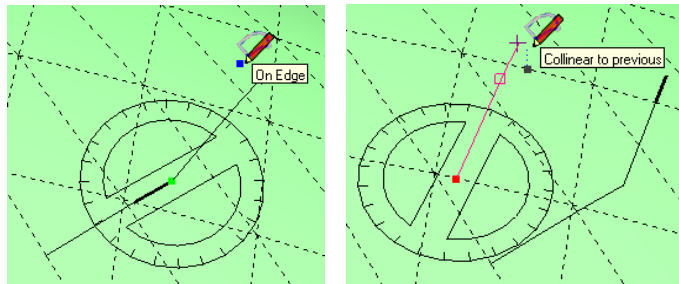
- ▪ **Behavior**
  - – **Click** on Origin, then **Drag** to End point and **Release**. You can also Click, Release, Drag and then Click again. Note that in the first mode, the script assumes you want to draw <u>disjointed</u> lines. In the second mode, it will <u>chain</u> lines (this behavior is strictly the same as for the native Sketchup Line tool).
  - – **Hit an Arrow key to force an axis direction**, and Arrow Down to go back to no particular direction.
  - – You can **type a distance in the VCB** (in current units, positive or negative) which **will prolong (or shorten) the line up to that distance. You can also type** an angle value (in degree, followed by **'d'**) to impose a direction.
  - – When lines are chained, Type **Escape** to start from a new Origin.
  - – **Redo by Double Click** is interpreted as using the previous distance if you already dragged the End point, or to draw a line with the same orientation and distance if you start from a new origin, before clicking (this latter option can be useful to draw parallel lines at different points).

- ▪ **Options**
  - – **Plain Lines versus Construction Lines**: **F2** or **Ctrl Alone**
  - – **Mark Vertices with Construction Points**: **F3**
  - – **Generate as Group**: **F6**
  - – **Activate Protractor**: **F7**
  - – **Generate Faces**: **F8** (Plain Line only)
  - – **Generate Contour as Curves**: **F9** (Plain Line only)

6. Shapes on Surface

The script provides a suite of Shapes, with some additional options:
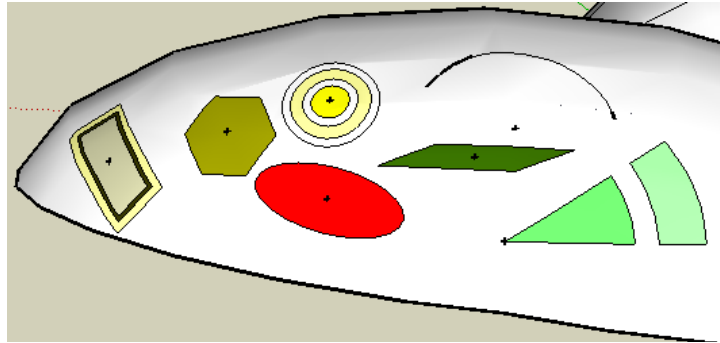
In version 1.5, you have
  - **Circle** and **Polygon**
  - **Rectangle**, defined by its length and width
  - **Parallelogram**, defined by its length, width and angle
  - **Ellipses**, defined by axes
  - **Circle** defined by 3 points
  - **Arcs** (circular)
 - **Sectors**, defined by origin, direction and angle

All shapes have in addition the option to draw *Rings*, that is, the same contour offset by a negative or positive distance. All generated contours can be later edited by the Contour Edition tool (see below).

1) Common Features to all Shapes

All shapes share some features and options summarized below:

a) Rings

Via the VCB, it is possible to specify the drawing of the same shape, offset by a distance, positive for outside and negative for inside. The syntax is a distance followed by 'x'. For instance, if you draw a circle of radius 100, then, by typing in the VCB, *at any time*, the string *"30x -10x -20x"*, you will generate the following shape:
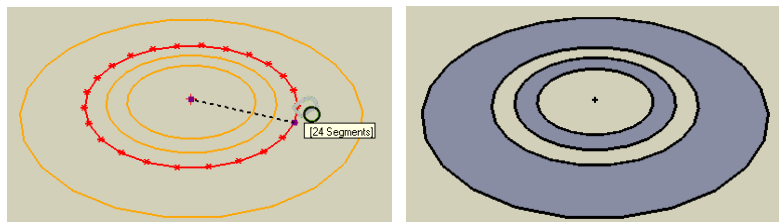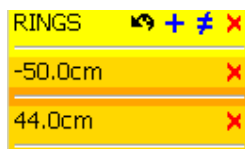
Note that the generation of faces tries to guess the result you would like to obtain, based on the even or odd number of rings.

In version 1.5, there is dedicated palette to enter and visualize Rings.

b) Axes dimensions

Shapes are defined by one dimension (polygon, circle) or 2 dimensions. Each shape has its proper definition of what are these 'dimensions". For a rectangle, it would be the length and width, for an arc it will be the chord and sagitta, etc….

What is useful to know is that:

- **Escape** allows moving back in the input states.

- **You can type the dimensions in the VCB**, as natural value (i.e. not followed by any character). If you input only one number, it would apply to the current dimension you are drawing. If you type two, it would normally impose the first and second dimensions and execute the shape drawing.

- **Orientation of directions for axes use the inference mechanism on surface**, as seen for the Line tool. For Sector, the Protractor is automatically displayed, but you can call it as well by pressing **F7** for parallelograms and Circle by 3 Points if you wish to.

- **When applicable, you can change the input mode to select the right meaning of axis by toggling F5**.

  – <u>For circles and polygons</u>, this would be by Radius or by Diameter.
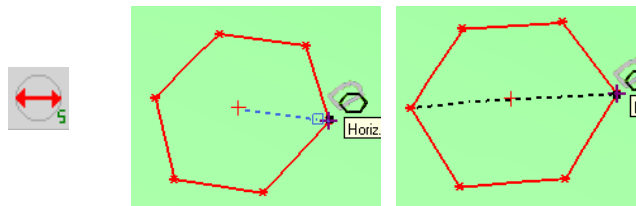
  – <u>For Rectangle, Parallelogram and Ellipses</u>, you have 4 modes, based on full or half length, and full or half width.
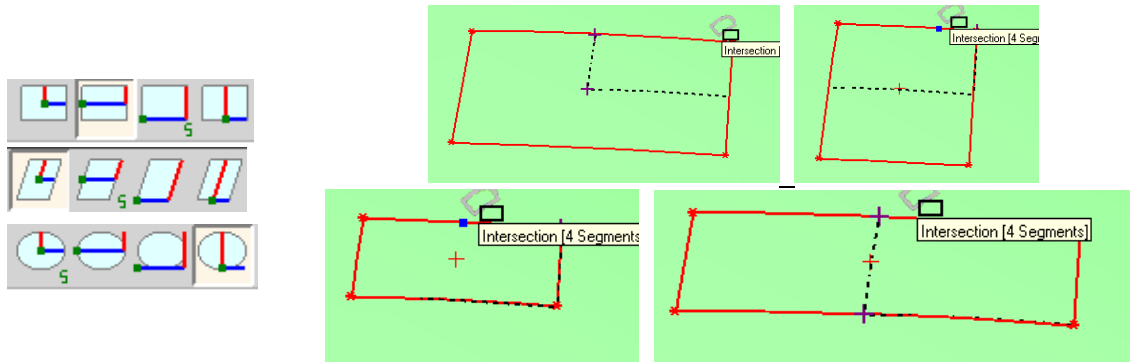
  – <u>For Sectors</u>, the angle is specified by reference to the Trigonometric sense, but F5 will flip the angle.
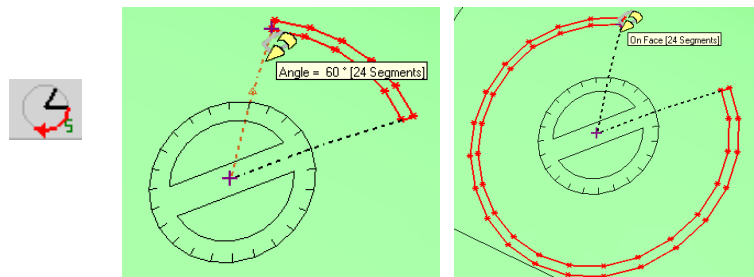
  – <u>Other shapes</u> (Arc, Circle 3 Points) have only one input mode.

c) VCB Syntax

Because shapes are all different, some parameters may or not apply, but here are the conventions:

– **You can specify the parameters at any time during drawing**, and **after** the shape has just been generated to modify it afterward.

– **You can type several parameters in the same command**. The usual separator is **'space'**, but **';'** is also valid[6]. The command is read from right to left and the latest value(s) are taken when redundant.

– **The possible parameters are**:

▪ **Dimensions**: just the number, in current units

▪ **Number of segments**: integer followed by **'s'**

▪ **Rings**: positive or negative dimension, followed by **'x'**. By convention **'0x'** will erase all rings (or just 'x', if there are other parameters before). The notation **'[a b c]x'** (or **'(a b c)x'**) allows to specify directly rings, deleting previous ones and replacing them by rings with offsets a, b and c.

▪ **Angle**: positive or negative value in degree, followed by 'd'.

For instance, assuming your current unit is 'cm', the following VCB string *"30 x 60d -2.4x 40 4x"* for a Sector will generate a sector of radius=40, an angle of 60 degree and 2 rings, one at -2.4 cm (inside) and one at 4cm (outside).

d) Redoing by Double Click

You can redraw the same shape elsewhere on the surface by a Double Click on the new origin. The new shape will have the same orientation and dimensions.

If you double click while drawing a shape, the shape will be generated with the missing parameters taken from the previous shape drawn.

### e) Drawing shapes on surface <u>without</u> a surface

As you can guess, when there is no surface, you need to figure out one. Sketchup has the same problem and usually uses the horizontal plane. For the Shape tool, this is the default, but you have the option to specify your plane direction:

▪ By typing an **Arrow key**, corresponding to the normal to the plane, so:

– UP: for horizontal plane
– LEFT: for vertical plane defined by Red and Blue axis)
– RIGHT: for vertical plane defined by Green and Blue axis)
– DOWN: custom plane (see below)

[6]   Note that both comma and point are reserved for decimal separator

- By selecting an existing face of the model and use it as the plane for drawing. For this, just mouse over the face, without **clicking and hit Shift.** The face will briefly highlight in blue. Then the shape will be drawn in the plane defined by your origin point and the plane of this highlighted face.
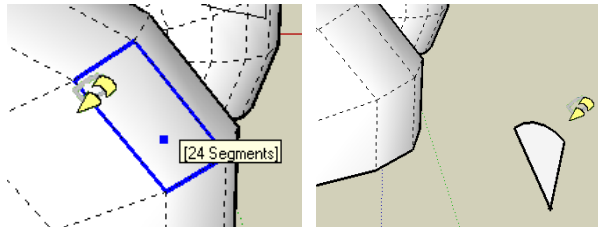

Important: The above applies to the 'default' plane when there is no surface. When drawing on a surface, the surface geometry will of course have precedence.

f) Generation of faces

The script may generate faces when appropriate, usually when the shape is not drawn on a curved face. When using rings, the generation of faces follows a logic of alternate faces from the outside toward the inside.
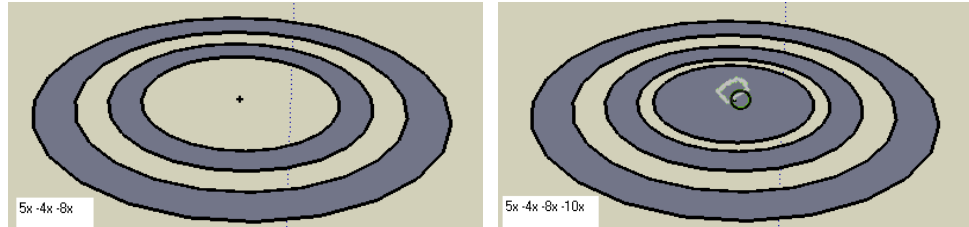
Sectors receive a special treatment, as the script also generates borders:
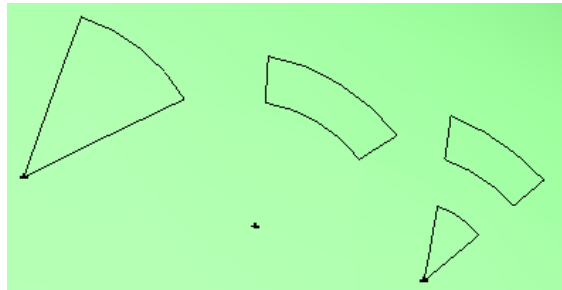
The generation of faces can be activated or deactivated by **F8**.

g) Other options

Shapes also support the usual options provided with other On Surface tools

- **F2 or Ctrl alone**, to draw in Construction line
- **F6** to generate in a Group
- **F3** to generate construction points at all intersections
- **F9** to generate or not the contour as curves

And, as seen above…

- **F5** to toggle applicable axes inputs
- **F8** to generate or not faces when possible

2) Rectangles, Parallelograms and Ellipses
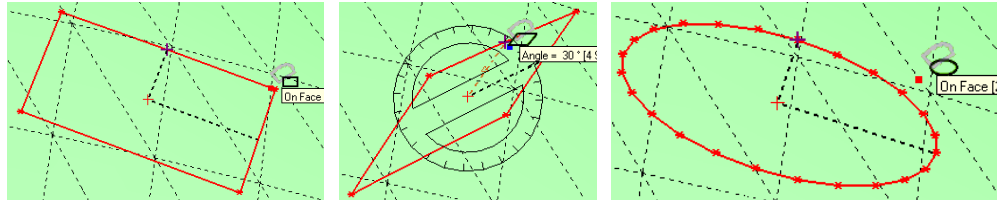
They all require:
- A direction, given by the first segment
- A first distance (length)
- A second distance (width)

I assume that, beyond quick trials, you will use it with given dimensions in mind, so that drawing the first segment imposes both the direction and the first length.

As said, you have 4 modes of inputs, combining length or ½ length, width or ½ width, that you can specify visually, using inferences, or via the VCB.

For Parallelograms, you can also specify the angle to orientate the second axis, via the protractor or via the VCB(Angle value followed by'd').

For Ellipses, you can specify the number of segments in the VCB

In all cases, remember that when you are on a curved surface, angles and distances are not always preserved exactly. So, don't expect you get a pure rectangle on some convoluted surface.

3) Circles and Polygon
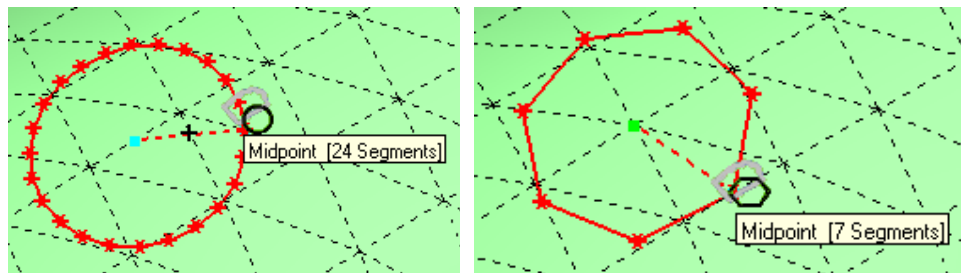
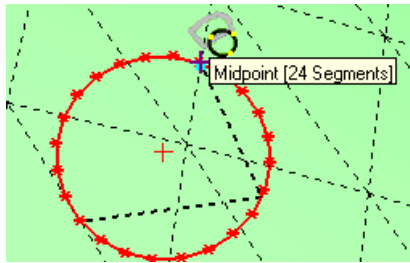Polygons and Circles are actually the same tool, but just differ by the default number of segments:
- **Circle** (default number of segments is 24)
- **Polygon** (default number of segments is 6)

You can have 3 to 150 segments, so from triangle to very high-poly circle!

They just require one dimension, the radius or the diameter (toggle **F5**).

4) Circles defined by 3 Points

This may be useful, as, on a surface, it would not be easy to construct manually.

You need to draw a first segment linking 2 points, and then a second segment for the third points. Otherwise, it is just a circle!

Note: you can change its number of segments, but don't expect any capability of trilateration when the number of segments is 3. The geometrical construction assumes that it is a circle with a high number of points, and does not force the second axis to end on a point of the generated segment. I may improve all that in a future version.

5) Arcs
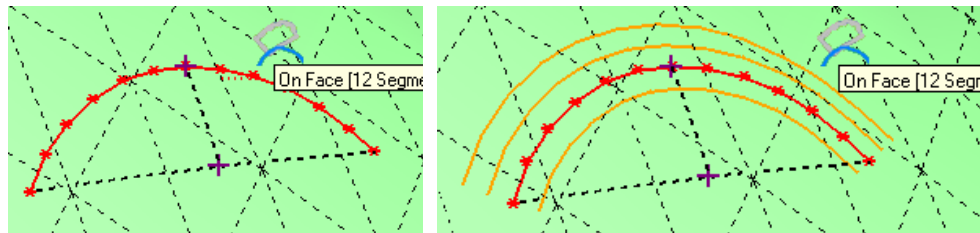
As in native Sketchup, you will first draw the chord and then the sagitta (perpendicular at the middle of the chord).

When using rings, there is no automatic creation of borders.

Note that th**e main issue is when the sagitta is very small. This means actually that the center of the arc is very far, and could be outside of the surface**. In such a case, the calculation is done differently, but is a lot less accurate. So unexpected results may occur.

Finally, I did not do yet inference when the edges of an arc are tangent to the bordering segments. But you should notice that on any curved surface, you cannot guarantee tangency at both ends.

6) Sectors
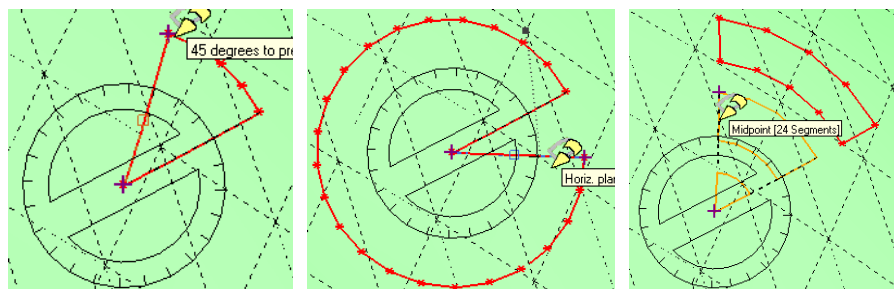
Sectors are defined by their center, a direction and then an angle. So it's pretty natural to draw. The convention for the angle is the trigonometric sense, but this can be reverted by toggling **F5**. The number of segments applies to the whole circle, not to the sector.

7. Freehand / Polyline on Surface

This tool was originally analog to the Sketchup Freehand tool, but drawing on curved surfaces. **As of version 1.5**, the tool has evolved to include a Point and Click mode, <u>with and without inference</u>, and thus offer a capability of *Polyline on Surface*.
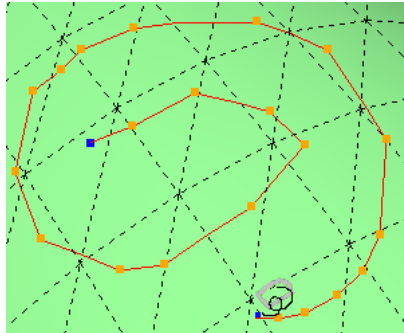
Within the same operation of contour drawing, you can switch between the different modes:

- **Auto-Sampling vs. Point & Click**, using the **F4** or **TAB** toggle key

- **With vs. Without inference**, using the **SHIFT** modifier

In Auto-sampling mode you just move the mouse to acquire points. The contour is finished when you click down (or release the mouse button). Note however that is you activate Inference, a double-click will be needed to finish, because a single click is precisely required to capture an inference point.

In Point & Click mode, you finish the contour by double-clicking on the last point (or selecting the 'Done' option in the contextual menu.

Although there is indeed some analogy with the native Sketchup, there are a few differences:

1) The tool supports 2 input modes, **Mouse button depressed** (like in Sketchup native), or **Mouse button released**. This second mode is, in my mind, more natural, and also offers more possibilities, as listed below

2) You can modify the sampling distance in pixel by typing it **in the VCB**. By default, it is 30 pixels, but it can be changed while drawing

3) You can also suspend the input of points by **keeping CTRL pressed** while moving the mouse, or by **toggling CTRL** to have a permanent pause (convenient for instance to orbit the model or change options in the button palette).

4) You can force an inference at any remarkable point of the model by **keeping Shift pressed**. Note also that the tool will automatically adjust the position of the input point if you reuse a point of the current contour (so you can close contour). If you **toggle SHIFT**, you activate Inferences permanently.

5) You can backtrack on the last points drawn by hitting **Escape**. When in Auto-Sampling mode, you need to move the mouse cursor close to the last point to resume drawing.

6) You can edit the created contour afterward to adjust or suppress vertices.

7) And of course, you can draw plain lines or construction lines, generate the contour in a group, generate it as a curve (default), and mark all intersections with a construction point.

There are some slight differences in the capture of points depending on the acquisition mode you choose:

- **Auto-Sampling mode**
    - **Click** on Origin, then **Drag** and **Release** when finished. You can also **Click, Release, Drag and then Click again** (much better mode). Inference is always active for the Origin, but not afterwards (except if you keep Shift depressed or work in Lock Inference mode).
    - **The capture of points works slightly differently depending whether you lock inference permanently or not** (state of the *'Locker'* button)**:**
        - **If OFF**
            - **To force an inference** (*at vertex*, *on edge*), **press SHIFT** and maintaining down until you see the mark of inference, **then release SHIFT**.gently to capture the point and then continue drawing.
            - **Single click** to finish
        - **If ON**
            - **To force an inference**, **click down** on the point point when displayed. If you don't, the inference point will be ignored.
            - **Double-click** to finish, or alternatively, pursue the drawing with the mouse button pressed, and finally release it to finish.
    - **In the VCB**, you can specify both the sampling distance in pixel by typing an integer.
    - To remove the last point (s), Type **Escape**. You will need then to move the cursor close to the last remaining point to resume drawing.

- **Point & Click mode**
    - **Click** on Origin, then **click the following points**.
    - **Double-click** to finish, or alternatively, pursue the drawing with the mouse button pressed, and finally release it to finish.
    - Type **Escape** to remove the last point(s), recursively.

- **Options**

    All options are available from the palette and contextual menu. Note that the green Check Mark can be used to terminate the drawing too.

    FREE HAND -- Click to pick points - Double Click to finish
    Edges = 0   Pixels 40

    - **Plain Lines versus Construction Lines**: **F2** or **Ctrl Alone**
    - **Mark Vertices with Construction Points**: **F3**
    - **Generate as Group**: **F6**
    - **Generate Faces**: **F8** (Plain Line only)
   - **Generate Contour as Curves**: **F9** (Plain Line only)

8. Eraser On Surface

This tool is more or less the replica of the standard Sketchup Eraser, but dedicated to Contours on Surface. As such, it only deals with these contours 'on surface' and will superbly ignore any other type of lines.

The tool supports separately plain lines and construction lines / points. Press **Ctrl alone** or **F2** to switch between the two modes.

After starting the Tool,

- **Mouse over contours**; edges which can be deleted are normally highlighted in orange.

- **Click and keep the button depressed** to select edges to be deleted.

- **When you release the mouse button**, the edges will be erased

- **Press Escape to ignore all edges already selected**

- If you keep **Shift depressed**, when selecting edges, you will actually **select all other edges connected** and forming a valid contour on surface. This option allows you deleting full contours in one click.
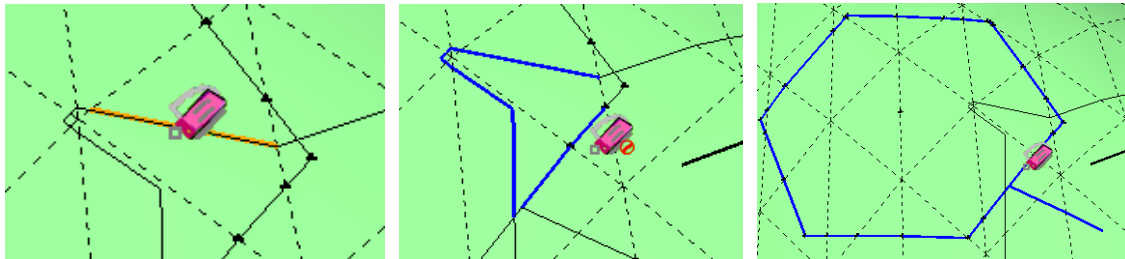

- **Options**

  No option, apart from toggle **F2** (or **Ctrl Alone**) between Line mode and Construction Line mode.

9. Edition of Contours on Surface

As drawing accurate shapes on curved surfaces can sometimes be shaky, I found useful to provide a tool to edit the contours on surface. There is no equivalent native tool in Sketchup, so here's how it works:

1) **Start the tool and move the mouse over the model** (no click or mouse button depressed). The tool will highlight contours as you pass over.
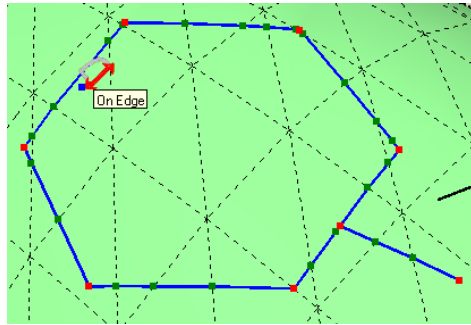
2) **Move a vertex**: **Click and drag** the vertex to its new position.

You will notice that the marks on the contours are **Green** and **Red**. The Red ones are called *Anchors* and usually correspond to the meaningful points of the contour (start and end points for lines, vertices for polygons and circles). The Contour editor allows you to move a vertex by reference to either the closest anchors or just the next points on each side. The convention is the following:

  – **If you move a Red vertex**, the reference is the two closest Red Vertices
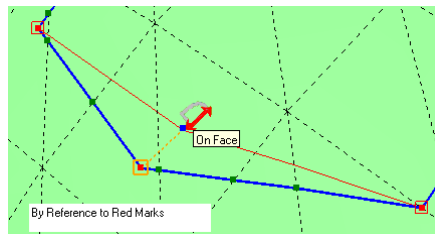
  – **If you move a Green vertex**, it will be by reference to the next points, whether Green or Red
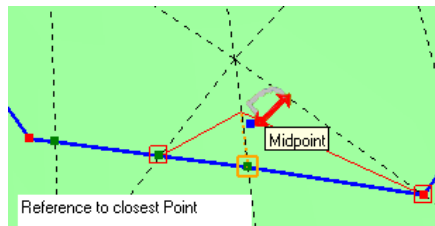
You can reverse this behavior by keeping **Ctrl Depressed**[7] while dragging the edited vertex.

In addition, you can **toggle the color of a mark** by clicking on it and keep the mouse down for at least 0.3 second (and then release).

Finally, note that when a vertex is moved, it becomes automatically Red, as the script assume that you want to give it a pivot position.

---

[7] In version 1.2, it was Shift, but this is now reserved to lock inference when moving the vertex (in a future version)

3) **Delete a vertex**: **Double-Click on the vertex**. This will remove it from the contour references. Note however that new vertices can be created as a result of this operation.
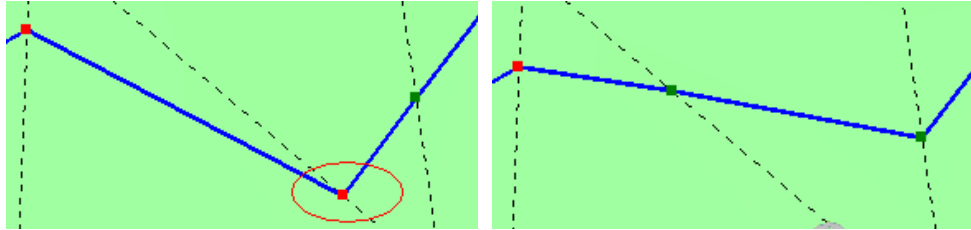
4) **Add a vertex on an edge**: **Double-Click on the edge** where you wish to add a new vertex, which you will then be able to move later on.
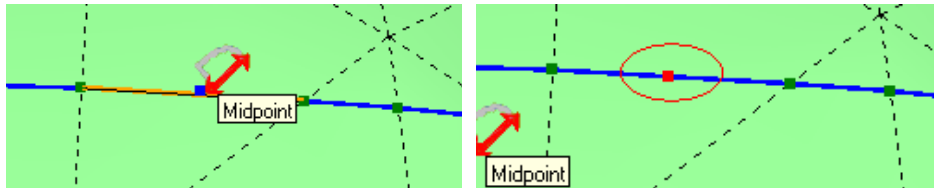
The script will create the intersection points when vertex are moved or erased.

Caveats:

- For segments of the contour that are collinear to pre-existing edges of the surface, the script tries to reestablish their initial properties (i.e. soft, smooth). This may not always work in some situations.
- Contours generated by Offset On Surface may have very few Red points, as the vertices of the original contour may have been 'simplified'.

10.   Known Problems

There are a number of issues with the scripts that I have been identified myself or that have been reported by users. I hope to fix them progressively or suggest workarounds.

1)   Hazardous contour on closed surfaces → still open

When you draw on the surface of a shape that is closed, the algorithm may lead to strange behaviors, especially when it passes over sharp angles. The reason is that on closed surfaces, there are many paths between two points.

I think I will fix it one day by filtering the paths based on a limit to the angles of faces.

2)   'Straight' lines between points → fixed in version 1.3

From version 1.0 to version 1.2, the algorithm was simply wrong! The effect was mainly noticed on sheared surfaces or some regular surfaces when drawing lines in oblique directions over long distance. Furthermore, the generated contour was not symmetrical and depended on which initial point was chosen.
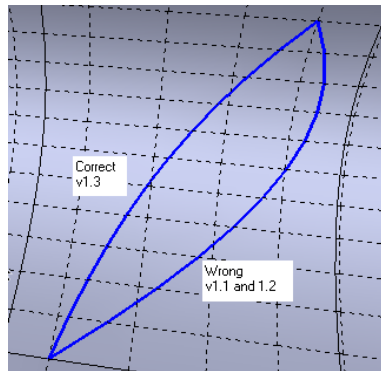
As of version 1.3, 'straight' lines are 'more' straight, and also independent of the first or second point chosen as origin.

3)   Contour Edition and faces → open

When you move a vertex of a contour, there is no mechanism to reconstruct the face(s) that are impacted. Note that, anyway, depending on which direction you move the vertex, you may not preserve a plane for the faces.
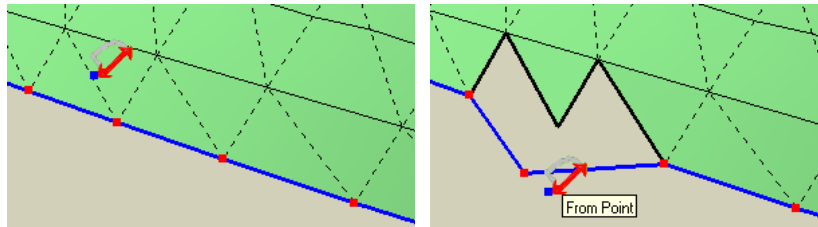
4)   Drawing Arcs → open

When you start drawing an arc, the center of the circle is actually very far from the chord. As the center is also supposed to be on the surface, it can even be on another side.On some surfaces, it