Sketcher

Starting from NaroCAD 1.6.0, the Sketcher concept was introduced. The main idea is to separate the 2D elements from 3D.

A sketch basically represents a bunch of 2D elements and relations that are contained in a plane. From the very beginning, we see that there are no 3D elements witch can be drawn into the sketch. This leaves a lower number of elements that a sketch can contain. To make the number of elements even lower, constraints will be used to define relationships between sketch elements. With the help of constraints, the rectangle shape will disappear. This means that a rectangle will only be defined as 4 lines with 2 parallel constraints, for example. Of course, we will offer more possibilities to build a rectangle. Even more the user should be able to build his own rectangle from scratch, just by using lines and constraints the way he sees fit. If we forget about the rectangle, triangle, and other shapes, we will only remain with points, lines, arcs, circles, ellipses and splines.

In order to draw the 3D elements, a few modeling commands will be used, commands tat will only work on sketches. Even more, one sketch will be affected by only such 3D modeling command. This way, the sketch will be designed to be as small as possible. So, a part will contain sketches and 3D operations performed on sketches.

The shapes that result from applying 3D command will have special dragging handlers, based on the sketch elements. For example, if a circle is extruded, the resulting shape must have a dragging handler in order to change the radius of the cylinder. Doing this will only change the radius of its corresponding element (the circle). With this logic, modifying 3D elements will only mean modifying the corresponding sketch elements and redrawing the resulting shape.

Now that we have an overview, let’s talk a bit more in detail.

As we saw above, the sketch is basically a collection of 2D elements and relations that are drawn on the same plane. This means that a sketch must be build on a plane. So, when starting the sketch command, NaroCAD will ask the user to select the defining plane. Clicking a face will start the sketch. Clicking on an empty part f the drawing or an invalid face will make the sketch use the XOY plane as the default support. When entering the sketch mode, the sketch specific tools will be enabled, and the tools that should work on the sketch will be disabled. This is a good approach because the user cannot make invalid operations, and the valid shapes will be much easier to spot. Similarly, the sketch elements will be disabled when exiting the sketch mode. Even more, a sketch with no elements will automatically be deleted. So, in the tree view, a sketch should hold its 2D elements and all existing relations. Some 2D elements should hold other 2D elements: for example the line, witch should hold 2 points as children.

Similarly, the relations will hold their respective children. Depending on their type, a relation can have one or more children.

The 2D elements:

Point

The sketch point is a 2D point that is defined on the current sketch. A point can be created by using the point command, or it might be generated automatically when creating a line.

Line

Similarly with the point, a line can be created from the line command, but also from the polyline or rectangle commands. Note that a line must generate its 2 base points, in case none of them are given as reference. Note that a line will always be defined with the help of 2 points: no more, no less.

Polyline

The polyline is similar with the line command. The only difference is that all lines except the first one will have the last line's endpoints as start point reference automatically set. For example, a polyline formed by 2 lines will have 3 reference points: 2 points for the first line, and one new point for the second line. Note that the polyline is not a new element. It is just a faster way to draw more lines. The fact the all lines starting from the second one have a reference set is not even defined as a relation. This is done from the fact that a line does not store hardcoded coordinates, but existing sketch elements.

Circle

The circle is defined with the help of 2 points: its center point and one point on the circle. When starting the circle command, the user will first define the center point, and one more point in order to define the circle radius. The circle should be moved by holding and dragging its center point, and resized by holding and dragging the other point. Such an implementation is necessary because we wouldn't want to move the circle by dragging one of the points defined on it. This is also important for the extrude editing handlers because the whole process will be reduced to dragging one of the points defined on the circle.

Arc

Drawing an arc will be done in more ways: by giving the center point, the radius and the end point, by giving the start point, the middle point and the endpoint, or other useful ways. This is similar with the line and poliline commands. Still, this is a bit trickier to make, because the arc only needs to store its center point, its start point and its endpoint. By drawing an arc with the help of the start point, midpoint and endpoint, the center point must be automatically calculated. This might be trickier to implement correctly, because the resulting arc will depend on the order of the given points. Calculating the center point and redrawing the arc might lead to the part of the arc that was originally missing. Similar with the circle, the arc should be moved with the help of its center point, and edited with the help of its start point or end point. The role of the start point is basically to determine the radius of the arc. This means that repositioning the start point should only modify the radius of the circle. In order to modify its length, the user should reposition the arc's endpoint. The endpoint should only be responsible to change the length of an arc, not to modify its center or radius.

Ellipse

The ellipse should be pretty straightforward: it should have a center and 2 other points that should determine its small and bigger radius. The functionality should be similar with the circle's.

Spline

The spline is defined by 2 or more points. In the near future, closed splines will also be available. Editing a spline basically means to move its control points.

Rectangle

The user should be able to draw a rectangle in more ways. One should be by giving 2 points. This way, a rectangle parallel with the plane's axes will be drawn, at the locations given by the 2 points: representing the diagonal of the rectangle. So, when drawing such a rectangle, 4 lines and 2 constraints (relations) should appear in the tree view. Of course, the user is able to make such a rectangle just by drawing 4 lines and applying 2 parallel constraints. Other types of rectangles might have somehow a center point defined. Even more ways to draw a rectangle can be added when needed. The main obstacle is to apply the constraints correctly, so that the rectangle should remain a rectangle, no matter how the user drags its helper elements.

Fillet/Chamfer

This command is quite tricky. It should generate an arc at the intersection of two lines. It is not as easy as it seems because an arc must be added between 2 lines, and the lines must be correctly redrawn. Even more, the arc must have its start and end point defined as the 2 lines start/end points. The trickiest part is to correctly remove parts of a line or other elements, and correctly add the arc element. Similar results should happen when applying chamfer. One tricky

Trim

This is one of the trickiest 2D commands. The main problem is that the circle must be transformed into an arc, and other 2D elements might be trimmed somewhere in the middle, and the remaining shapes should behave like one. For example, if a center part of a line is trimmed, its extensions must behave like a single line. A solution to this problem is to add the correct constraints after trimming the desired part of the line. The sketcher is a great help for the trim command because the number of elements in a sketch should be small. This automatically brings a great performance bonus.

Mirror

The mirror command should be able to mirror elements along points or lines. If we think careful about it, every 2D shape has point children. This means that the mirror command must generate correct mirrored points and use them as children for the mirrored elements. Just as fillet, the mirror command just needs to add a few 2D elements with the correct constraints.

Helper elements

This command can be applied on 2D elements in order to tell NaroCAD witch geometry should be used for when 3D command are applied on the sketch. Helper geometry will only be visible inside the sketch. The role of the helper geometry is to create certain behavior between certain sketch elements. For example, a hexagon could be build with a circle as helper geometry and a bunch of lines that are tangent to the circle. Without the tangency, the angles between the lines would not be correctly defined.

Elations between sketch elements (constraints)

Fixed

The fixed constraint has the purpose to make a shape stay in the same location. A point with the fixed constraint cannot be moved from its place. This is helpful when a sketch is built on an existing surface, because moving it will cause unexpected results. This constraint can be implemented as a certain horizontal distance from the sketch center and a certain vertical distance from the sketch center.

Vertical/Horizontal distance

Thee constraints calculates the vertical/horizontal distance between 2 points and makes it a constant value. This constraint is useful when certain 2D elements must have a certain distance between them, or when trying to fix an element to a certain position. Using these 2 constraints could be better in certain situations, because the user can see the constraint in the drawing area and edit it as he sees fit.

This constraint can be applied on points.

Vertical/Horizontal

The vertical/horizontal constraints have the role to make certain elements horizontal or vertical. These constraints should obviously appear when selecting a line. But 3 points, for example, can also be made horizontal or vertical to each other. So, these constraints do not work necessarily only on line elements.

Perpendicular

This constraint can only be applied when 2 lines are selected. The 2 lines are made perpendicular to each other.

Parallel

The parallel constraint works similar with the perpendicular constraint, when selecting 2 lines. The 2 lines are made parallel to each other.

Intersect

The intersect constraint can be applied when 2 elements are selected. The elements must intersect each other. Applying this constraint on 2 points should basically add a zero vertical and horizontal constraints between the two. When applying this constraint on a line and a point, the point may also be positioned on the continuation of the line. The same goes for an arc and a point.

Tangent

This constraint can be applied on a line and another 2D element like: circle, arc, spline and ellipse.

More about the sketch

Note that the sketch is defined on a plane and has a zero point. It is also important to notice that sketch elements should be able to detect other elements that are draw in the same plane, or 3D shapes witch have important elements on the current sketch. This behavior greatly enhances the program's capabilities. As we saw above, the sketch is made so that only one 3D operation can be applied to it. Keeping a few elements on each sketch will make the constraint to work fast enough even when the drawing is very complex. Not using the sketcher system, and calculating the constraints on every element of the drawing would make NaroCAD slow.