



Fundamentals of StudioTools

Copyright and trademarks

Software copyright information is located in the application, and can be accessed from the menu by choosing **Help > About StudioTools**.

All documentation ("Documentation") is copyrighted © 2001-2005 Alias and contains proprietary and confidential information of Alias. The Documentation is protected by national and international intellectual property laws and treaties. All rights reserved. Use of the Documentation is subject to the terms of the license agreement that governs the use of the software product to which the Documentation pertains ("Software"). The authorized licensee of the Software is hereby authorized to print no more than one (1) hardcopy of any Documentation provided in digital format per valid license of the Software held by such licensee. Except for the foregoing, the Documentation may not be translated, copied or duplicated in any form (physically or electronically), in whole or in part, without the prior written consent of Alias.

Alias and the swirl logo, Maya and DesignStudio are registered trademarks and Alias Natural Phenomena, Alias OpenAlias, Alias OpenModel, Alias PowerCaster, Alias PowerTracer, Alias RayCasting, Alias RayTracing, Alias SDL, ImageStudio, Alias Spider, StudioPaint, StudioViewer, StudioTools and SurfaceStudio are trademarks of Alias Systems Corp. ("Alias") in the United States and/or other countries. Silicon Graphics, SGI and IRIX are registered trademarks and Inventor is a trademark of Silicon Graphic, Inc. in the United States and/or other countries worldwide. Microsoft and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Renderman is a registered trademark of Pixar Corporation. Apple, Quicktime and Macintosh are trademarks of Apple Computer, Inc. registered in the United States and other countries. Adobe, Postscript and Illustrator are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Unigraphics, NX, and I-deas are registered trademarks or trademarks of UGS Corp. or its subsidiaries in the United States and in other countries. Arius3D is a registered trademark of Arius3D Inc. Cyberware is a registered trademark of Cyberware Laboratory Inc.. Cyrax is a registered trademark of Leica Geosystems HDS Inc. Steinbichler is a registered trademark of Steinbichler Optotechnik GmbH. Autodesk and AutoCAD are either registered trademarks or trademarks of Autodesk, Inc./Autodesk Canada, Inc. in the USA and/or other countries. CATIA is a registered trademark of Dassault Systèmes S.A. PTC, Pro/ENGINEER and Granite are trademarks or registered trademarks of Parametric Technology Corporation or its subsidiaries in the U.S. and in other countries. Further information about the GNU Lesser Public License may be found *here*. All other trademarks mentioned herein are the property of their respective owners.

All PTC Technology logos are used under license from Parametric Technology Corporation, Needham, MA, USA.

Not all features described are available in all products.



Alias Systems Corp., 210 King Street East, Toronto, Canada M5A 1J7

Contents

Introduction vii

StudioTools Concepts 1

Background 2

Points 3
History of splines 4
Mathematical representations of curves 5
NURBS 8

Curves 9

CVs, hulls, and edit points 10
Moving edit points vs. moving CVs 12
Multi-knots and CV multiplicity 13
Rational vs. non-rational geometry 14
Constructing quality curves 16
Blend curves 18
Keypoint curves 20

Surfaces 21

Isoparametric curves 22
Patches 24
What NURBS surfaces can't do 25
Curves-on-surface 26
Trimming 27
Shells 28

Object properties 30

Degree 31
Parameters and parameterization 32
Normals 35
Pivot points 36
Construction history 37

Modeling concepts 38

- Absolute and relative addressing 39
- Momentary and Continuous buttons 41
- Curvature 42
- Laying out curves and surfaces 43
- Continuity 47
- The construction plane 50
- Dynamic Shape Modeling 51

Meshes 62

- What is a mesh? 63
- Difference between meshes and polysets 64

Introduction 67

- What is animation? 68
- What can you animate? 69
- Basic workflow for manually creating an animation 70
- What happens when an item is animated? 72
- How can I tell if something is animated? 75
- What is a parameter curve action and a motion path action? 76
- What happens when you animate a camera on a curve? 79
- Can I reuse animation on another channel? 81
- What is inverse kinematics? 82
- What is a time warp curve? 84

Rendering 87

- The rendering workflow 88
- Shaders 89
- Shading models 90
- Textures 91
- Rendering methods 92

Introduction 93

- Introduction to Data Transfer 94
- Learn how Solid Modeling Theory works 96
- Learn the Solid Modeling workflow 98
- Learn about the tolerance requirements for Solid Modeling 99
- Learn how to get the topology right before transferring data 101

Requirements and workflows for CAD packages 102

- Pro/ENGINEER 103
- CATIA V4 111
- CATIA V5 114
- I-deas NX series 117
- Unigraphics 121
- Solid Imaging 124

TC VisProducts 127

Index 129

Introduction

Welcome to StudioTools Concepts.

This book has been assembled for your convenience from the concept sections (“About” information) of all the other books that describe how to use the interactive StudioTools products. Most of the information in this book doesn’t rely on you having the application available in front of you, so you may want to print this book to read at your leisure.

StudioTools Concepts

Explains basic concepts and terminology used in the StudioTools interface.

Background

Explains the origin and meanings of some of the basic concepts used in StudioTools.

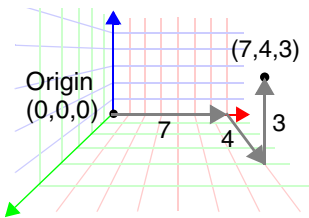
Points

A point is a location defined by three spatial coordinates. It has no size.

The most basic visual entity is the *point*. The point has no size, but it has a *location*.

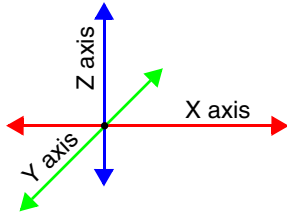
To determine the location of points, we first establish an arbitrary point in space as the *origin*.

We can then say a point's location is so many units left (or right) of the origin, so many units up (or down) from the origin, and so many units higher (or lower) than the origin.



These three numbers give us the *3D coordinates* of the point in space. For example, a point 7 units *right*, 4 units *down*, and 3 units *above* the origin has the 3D coordinates $(7,4,3)$.

To specify points on the opposite side of the origin, we use negative numbers. In the example, a point at $(-5, -2, -1)$ would be 5 units *left* of the origin, 2 units *up*, and 1 unit *below*.

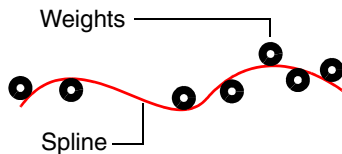


In computer graphics, we don't really say the point is "left/right", "up/down", or "higher/lower". Instead we call the three dimensions the *X axis*, the *Y axis*, and the *Z axis*.

History of splines

Describes the history of the representations of curves, from shipbuilding to modern computer modeling.

Splines are types of curves, originally developed for shipbuilding in the days before computer modeling. Naval architects needed a way to draw a smooth curve through a set of points.



The solution was to place metal weights (called *knots*) at the control points, and bend a thin metal or wooden beam (called a *spline*) through the weights.

The physics of the bending spline meant that the influence of each weight was greatest at the point of contact, and decreased smoothly further along the spline. To get more control over a certain region of the spline, the draftsman simply added more weights.

This scheme had obvious problems with data exchange! People needed a mathematical way to describe the shape of the curve. *Cubic Polynomials Splines* are the mathematical equivalent of the draftsman's wooden beam. Polynomials were extended to *B-splines* (for Basis splines), which are sums of lower-level polynomial splines.

See *Mathematical representations of curves* on page 5.

Then B-splines were extended to create a mathematical representation called NURBS, which are used by StudioTools.

See *NURBS* on page 8.

Mathematical representations of curves

Explains the mathematical basis of the curve representation used by StudioTools.

Polynomial equations

Starting with the simplest mathematical representation, we all remember from geometry class that we can represent a (two dimensional) line with an equation like $y = 2x$. For each value of x , we multiply it by 2 to get the value of y , and plot the two values on a graph.

The generalized form of this type of equation is $ax + by = c$. The expression to the left of the equals sign is called a *polynomial* ("poly" means many. It refers to the fact that the expression has more than one term).

We can make more complicated expressions where x is multiplied by itself, as $y = x * x * x$. Instead of writing out all the x 's in a term, we usually just count them and write the count as a superscript. The superscript is called "the exponent". So the expression above is written as $y = x^3$.

We can write polynomials with exponents, such as:

$$y = ax^2 + bx + c$$

(You may recall from math class that this is a *quadratic* equation). The exponent (the 2) on the first occurrence of x means that the graph of this function is curved rather than straight.

Degree

The *degree* of a polynomial equation is the largest exponent in the equation. Recall that the largest exponent on the equation for a line was 1. (When a term has no visible exponent, that is the same as an exponent of 1.)

- The degree of a linear equation is 1.
- A quadratic equation, which has a term x^2 , is degree 2.
- A cubic equation, which has a term x^3 , is degree 3, and so on.

Parametric representations

There are two general ways to write an equation for a curve. The *implicit* representation combines every variable in one long, non-linear equation, such as:

$$ax^3 + by^2 + 2cxy + 2dx + 2ey + f = 0.$$

In this representation, to calculate the x and y values to plot them on a graph, we must solve the entire non-linear equation.

The parametric representation rewrites the equation into shorter, easily solved equations that translate one variable into values for the others:

$$x = a + bt + ct^2 + dt^3 + \dots$$

$$y = g + ht + jt^2 + kt^3 + \dots$$

Using this representation, the equations for x and y are simple. We just need a value for t , the point along the curve for which we want to calculate x and y .

You can visualize parametric curves as being drawn by a point moving through space. At any time t , we can calculate the x and y values of the moving point.

This is a very important point, because the concept of associating a parameter number with every point on the line is used by many tools. This corresponds to the *U dimension* of the curve.

Creating complex curves

The lower the degree of a curve equation, the simpler the curve described. What if we want to represent complex curves? The simple answer might be to increase the degree of the curve, but this is not very efficient. The higher the degree of the curve, the more computations are required. Also, curves with degree higher than 7 are subject to wide oscillations in their shape, which makes them impractical for interactive modeling.

The answer is to join relatively low-degree (1 to 7) curve equations together as segments of a larger, more complex composite curve. The points at which the curve segments, or *spans*, join together is called an *edit point*.

Degree 5 and degree 7 curves are only available in some products or as purchasable options.

Higher degree curves should not be completely discounted, however. Degree 5 and 7 curves have certain advantages such as smoother curvature and more “tension”. They are often used in automotive design.

Smooth joins

A type of curve developed in the auto industry and familiar to anyone who works with common illustration programs is the *Bezier* curve. Bezier curves combine cubic curve segments, each with four control points (the start and end points, and two “handles”). The problem with Bezier curves is that the joins between segments are not necessarily smooth.

The solution to this problem, used by NURBS, is to use the last control points of the previous span as the first control points of the current span. This ensures smooth joins between curve segments. (Bezier curves can still be simulated perfectly using NURBS curves with multi-knots).

The degree of the curve determines the smoothness of the joins between spans. Degree 1 (linear) curves give positional continuity at the join. Degree 2 (quadratic) curves give tangent continuity. Degree 3 (cubic) curves give curvature continuity.

NURBS

Describes the meaning of NURBS, the curve and surface representation used for modeling in StudioTools.

NURBS stands for *Non-Uniform Rational B-Splines*.

- *Non-Uniform* refers to the *parameterization* of the curve. Non-Uniform curves allow, among other things, the presence of multi-knots, which are needed to represent Bezier curves.
- *Rational* refers to the underlying mathematical representation. This property allows NURBS to represent exact conics (such as parabolic curves, circles, and ellipses) in addition to free-form curves.
- *B-splines* are piecewise polynomial curves that have a parametric representation.

For more information on NURBS objects, see the following:

Curves (page 9)

Surfaces (page 21)

Object properties (page 30)

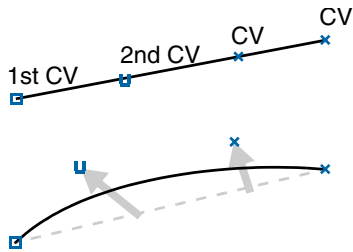
Curves

Describes curves as they appear in the StudioTools interface.

CVs, hulls, and edit points

Explains the origin and use of the different curve features.

CVs



CVs (*control vertices*) control how the curve is “pulled” from a straight line between edit points. They are the most basic and important means for controlling the shape of a curve. Lines between consecutive CVs form the control *hull*.

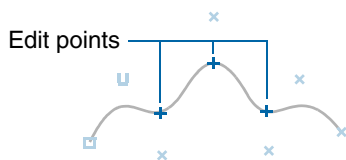
You cannot add CVs to the interior of a curve: there is always a set number of CVs for each span. The number of CVs is equal to the degree of the curve plus one. So, for example, a degree 3 curve has four CVs per span.

StudioTools draws CVs differently to let you tell the difference between the start and the end of a curve. The first CV (at the start point of the curve) is drawn as a box. The second CV is drawn as a small “U”, to show the increasing U dimension from the start point. All other CVs are drawn as small X’s.

Multiple spans

Longer and more complex curves require more than a single span curve. As you draw what appears to be a single long curve, StudioTools is actually adding several curve spans together. The last CV of the previous curve span become the first CV of the next curve span, creating very smooth transitions between the curve segments.

Edit points



You can tell when a curve is made from multiple spans in several ways. One is to look for *edit points* on the curve. Edit points mark the connection point between two spans. StudioTools draws edit points as small crosses.

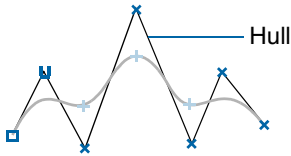
Unlike the on-curve control points of Bezier curves (used in many 2D illustration programs), NURBS edit points are not usually used for editing curves. CVs control the shape of a NURBS curve, and edit points are just indicators of how many spans a curve has.

There are, however, a few tasks that use edit points:

- If you want more control in a curve, you can insert an edit point to increase the number of spans in the curve and give you more CVs to work with.
- You can also delete edit points to decrease the number of spans in a curve (and probably change the shape of the curve).
- It is possible to move edit points to change the shape of a curve, but you should avoid doing this except for minor adjustments.

StudioTools does not actually move the edit point itself, but instead moves the CVs to reshape the curve so the edit point is where you specified.

Hulls



As a curve gets more spans/edit points, you might lose track of the order of the CVs. To show the relationship between CVs, StudioTools can draw lines between them. These lines are called *hulls*.

(StudioTools also provides other feedback to show the order of CVs. For example, when you pick a CV, StudioTools highlights its span within the curve.)

Moving edit points vs. moving CVs

Describes why moving CVs is preferable to moving edit points when reshaping a curve.

In theory, moving edit points would be an excellent way to edit a curve, since they lie on the curve itself. Unfortunately, it doesn't work out that way. This is because the shape of the curve determines the positions of edit points, *not the other way around*.

StudioTools *does* allow you to move edit points by "reverse engineering" the curve from the edit point. When you move an edit point, the **Move** tool tries to find a curve which passes through the new edit point location. Because this process is time-consuming and has an infinite number of solutions, the tool must place constraints on how moving the edit point affects the curve.

Because of these constraints, you usually cannot make major changes well by moving edit points. Moving edit points is best for small scale reshaping.

Even though it is slightly less intuitive, the only way to reshape the curve with complete power is by moving CVs.

Multi-knots and CV multiplicity

Describes two ways of achieving sharp bends in NURBS geometry. While these features are supported to a large degree in StudioTools, they can cause problems with certain tools and other software.

A multi-knot is multiple edit points at the same location in space.

CV multiplicity is multiple CVs at the same location in space.

Multi-knots are usually the result of curve or surface editing operations that require a sharp turn in a curve. CV multiplicity is created by manually placing adjacent CVs in the same location (using the **Magnet** tool).

Multi-knots and CVs with multiplicity are generally undesirable. Some tools (such as **Rail Surface**) cannot work with them, and many CAD packages will not accept models with multi-knots.

Multi-knots and CV Multiplicity achieve similar effects, even though they are different mathematically.

Multi-knots and continuity

Multi-knots eliminate one level of automatic continuity for each extra edit point.

For example, a degree 3 curve normally has curvature continuity (G2) at edit points.

- If you create a multi-knot of two edit points, you lose automatic curvature continuity, so you only have tangent continuity (G1) at the multi-knot.
- If you create a multi-knot of three edit points, you lose both automatic curvature and automatic tangent continuity, so you only have positional continuity (G0) at the multi-knot.



Only the intrinsic continuity is lost. As with Bezier curves, clever placement of CVs can restore continuity.

StudioTools only creates *full multiplicity* knots, i.e. knots which have a multiplicity equal to the degree of the curve.

Rational vs. non-rational geometry

Explains the differences and pros and cons of rational and non-rational geometry.

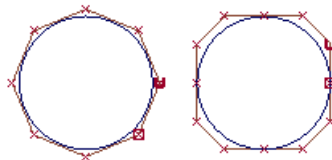
Non-rational geometry is a sum of polynomials. Rational geometry is a *ratio* of sums of polynomials. Rational geometry is considerably more complex mathematically. Therefore:

- It may not be transferable to downstream CAD packages that can't deal with complex descriptions
- It can be slower to manipulate when modeling, and slower to render.

The following tables lists the differences between the two types of geometry.

Nature	Pros	Cons
Non-rational	<ul style="list-style-type: none">• More flexibility for transformations.• Faster.	<ul style="list-style-type: none">• Sacrifices some precision for modeling flexibility.
Rational	<ul style="list-style-type: none">• Precise geometry (that is, exact conics).	<ul style="list-style-type: none">• Weighted CVs not supported by many CAD packages.• Weighted CVs harder to manipulate.• Creates multi-knots.• Slower to display and render.

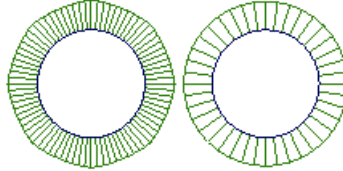
This illustration shows two circles drawn with the two types of geometry.



- The circle on the left is a non-rational curve with CVs that are all weighted equally. To have a non-rational curve, all weights must be 1.0.
- The circle on the right is a rational curve with different weights applied to the CVs, and multi-knots.

You can see the difference in two ways:

- If you attach a radius measurement to the circles, you will see that the non-rational circle is not a perfect circle (although it is pretty close): it has different radii depending on where you measure. The rational circle is a perfect circle.
- Attach curve curvature combs to the circles. The curvature on the non-rational circle on the left varies. The curvature of the rational circle on the right is constant.



Constructing quality curves

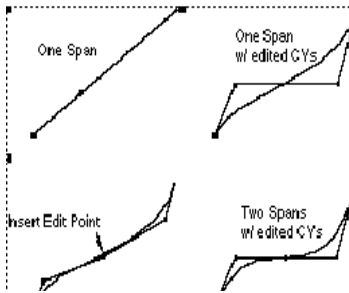
Contains tips for constructing curves that will make building high quality surfaces easier.

To create quality surfaces you need quality curves. These guidelines will help you create good curves.

Simple curves

Use the simplest curves that can describe the shape you want. Simpler curves mean simple, faster rendering surfaces.

One effective method for achieving simple curves is:



- 1 Begin a curve by drawing a single span.
- 2 Move the CVs to achieve the shape you want.
- 3 If you can't achieve the shape, add an edit point to create more CVs.
- 4 Continue until you have the shape you need.

This iterative process ensures your curve only has as many spans as are absolutely necessary.

You can also use the **Rebuild curve** tool to simplify existing curves. The tool can simplify a curve while maintaining its shape within a tolerance you set.

Parameterization

It is often best to build curves with uniform parameterization, because it makes inserting edit points and detaching curves at exact locations easier.

- When drawing Edit point curves with Uniform parameterization, the resulting CVs may be placed awkwardly. To fix this, move the CVs to prevent crossing hull lines.
- Try to consistently use either Uniform or Chord length parameterization when drawing curves. If you mix and match curve styles, it could result in cross knot insertion when the curves are used to build a surface.

Intersections

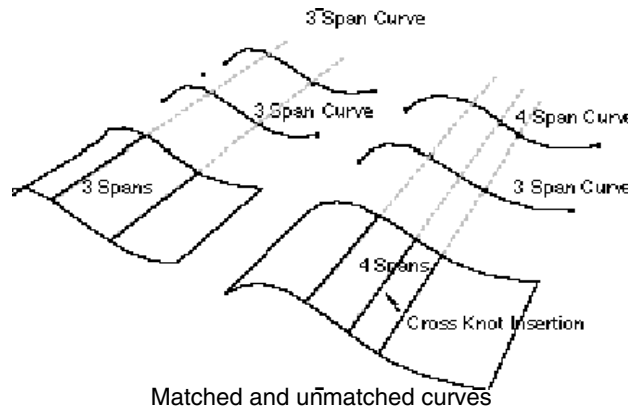
Some surfacing tools require curves to intersect:

- To draw intersecting curves, use curve snapping (hold down **Ctrl + Alt**, or click the curve snapping button **crv** to the right of the prompt line).
- To make existing curves intersect:
 - ◆ Pick an edit point and use the **Move** tool with curve snapping.
 - ◆ Use the **Object editor** with curve snapping.

Planning for surfaces

When creating curve, plan ahead to the surfaces that you want. Try to have the same number of spans in all the construction curves for building a surface. A simple way to achieve this is to start with one curve, then duplicate it to create more construction curves.

When you create a surface from curves with different numbers of spans, the new surface will have an extra isoparametric curve corresponding to every extra edit point. This is known as *cross knot insertion*. It makes the new surface more difficult to edit and more complex.



Blend curves

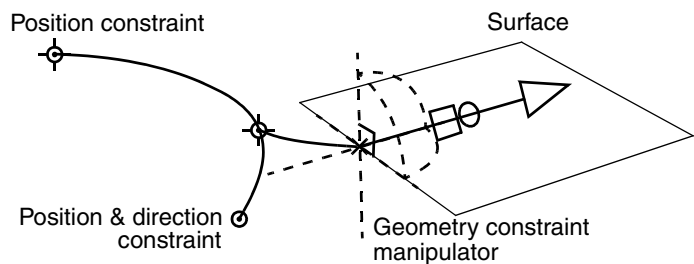
Describes the features and concepts behind blend curves, which allow you to create curves by specifying constraints on their shape.

Palette tool: **Curves > Blend curve toolbox**

Blend curves provide higher-level, simpler methods for shaping and manipulating curves. They provide a level of abstraction on top of the actual geometry of the curve. Blend curves let you focus on what the curve needs to do, and have the system calculate the right curve to fulfill those requirements.

Blend curves are normal NURBS curves with more construction history: you can use all the normal curve tools on blend curves, and when you are not using blend curve tools, they look like any other curve.

Blend curves are controlled by *blend points* acting as constraints:



You create the curve by setting up the constraints, such as

- what points in space the curve should pass through,
- which surfaces it should be tangent to,
- which existing curves the blend curve should intersect,
- what direction it should be travelling at a certain point,

...and so on. StudioTools draws the curve to satisfy the constraints, and automatically updates the curve when the constraints, or the objects the curve is constrained to, change.

Types of blend points

There are three main types of blend points.

- *Location*: forces the curve to pass through the blend point's location in space.







This is the type you create when you first draw a blend curve.

- *Direction*: forces the curve to pass through the blend point's location in space travelling in a certain world space direction.

There are two sub-types of direction:

- ◆ *Directed*: you set an actual direction for the curve tangent. Use this type when the specific tangent direction at the point of the blend point is important.
- ◆ *Parallel*: you set a line along which the curve passes (in either direction) at the blend point. This is easier to enforce and results in better curve continuity.
- *Geometry*: forces the curve to pass through a point on a curve or surface and travel in a direction relative to that curve or surface.

The following table shows the icons used to represent the different constraints:

Type	Not attached	Attached to blend curve	Attached to regular curve
Location			
Direction			

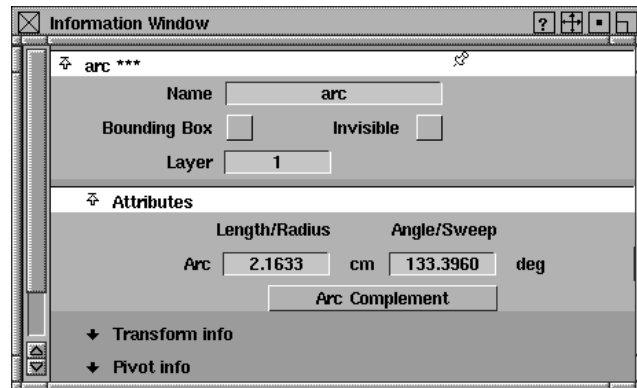
Keypoint curves

Describes the concepts behind keypoint curves, which allow you to create CAD-like lines and arcs.

Overview

- Keypoint curves retain more information than other curves. They remember relationships and constraints, and apply them when you edit the lines. You can also edit these special attributes in the **Information Window**.

For example, a keypoint arc has edit points and CVs just like a normal curve, but it also has a *radius*, *sweep angle*, and *center point*, all of which can be edited. During editing, the arc *stays* an arc: it will not lose its shape from keypoint editing.



When you combine keypoint curves into composite curves (for example, with the **Line-arc** tool), relationships between the individual lines and arcs are still maintained.

- Keypoint curve tools create *guidelines*, which are very useful for aligning curves with each other as you draw.
- Keypoint curves are especially useful for CAD and drafting applications. However, any part of your model requiring geometric accuracy or ease of editing will benefit from keypoint curves.



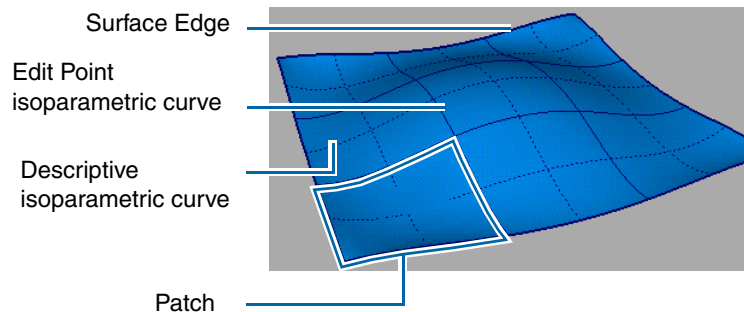
Most tools that work on normal curves also work on keypoint curves.

Surfaces

Describes how isoparametric curves, U and V coordinates, and possible trims combine to form a surface.

Isoparametric curves

Isoparametric curves are line running along the surface in the U and V directions, showing the shape of the surface as defined by the CVs.



StudioTools draws a NURBS surface as a mesh of curves, called *isoparametric curves*, running in the U and V directions.

Isoparametric curves are sometimes called *isoparms*.

Unfortunately, the term “isoparametric curve” is used to describe two related but subtly different features of a surface:

Edit point isoparametric curves

A line of constant parameter *at an edit point*. The isoparametric curves at edit points are special, since they represent the boundaries between “patches”. Like CVs, these isoparametric curves are important in representing the surface within the system.

StudioTools draws these types of isoparametric curves using **solid lines**.

- This is the type of isoparametric curve created by the **Insert** tool. Adding this type of isoparametric curve actually changes the geometry of the surface.
- You can only delete an isoparametric curves of this type.
- Using this definition, a surface has the same number of isoparametric curves in the U and V directions as it has edit points.

Descriptive isoparametric curves

Any line of constant parameter in either U or V. For example, if you join together every point on the surface where $U=1.5$, the resulting line is a U isoparametric curve:

StudioTools draws these types of isoparametric curves using **dotted lines**.

- You can increase the number of this type of isoparametric curve that is drawn for a surface with the **Patch precision** tool.
- Using this definition, a surface has an *infinite* number of isoparametric curves.
 - ◆ You can use these isoparametric curves to help you understand the surface shape, but the system doesn't use them to represent the surface internally.

Patches

Patches are the regions between adjacent edit point isoparametric curves.

The four-sided regions between adjacent edit point isoparametric curves or edges are called *patches*.

You rarely need to think about patches, since the focus in StudioTools is on the isoparametric curves.

One tool that works with patches is the **Patch precision** tool, which sets how many U and V isoparametric curves are drawn for each patch.

What NURBS surfaces can't do

Describes the fundamental limitations imposed by the geometry of NURBS surfaces and how to work around them.

Because of the underlying representation of NURBS surfaces, there are some things they cannot model:

- Topologies that are not equivalent to a rectangular sheet.
Spheres, cones, tori, and triangles can all be built from sheets by attaching or collapsing sides. But more complex shapes, for example a star shape, cannot be represented with a simple NURBS surface. To get a complex surface outline, you must use a trimmed surface or a network or collection of four-sided surfaces.
- Holes.
To create a hole in a surface, use a trimmed surface.
- Surfaces that cannot be mapped with regular U and V coordinates.
For example, you can model the shape of a Mobius strip, but the surface will have a seam.

Curves-on-surface

Curves-on-surface are special curves that exist on a surface, and are used mostly for defining the line along which to trim the surface.

Curves-on-surface are special curves that are drawn in the UV space of a surface, rather than in the XYZ space of the scene. Curves-on-surface do not have CVs. They are controlled by moving on-curve edit points.

You can create curves-on-surface by drawing directly on the surface, by *projecting* existing curves onto a surface, and by *intersecting* existing geometry with a surface.

Curves-on-surface are usually used to trim surfaces, or to form the edge of new surfaces.

Trimming

Describes the process of trimming, through which you can alter the visible shape of a surface by trimming away parts.

Since NURBS surfaces are intrinsically four-sided and do not allow holes, you need a way to visually simulate irregular shapes and holes when using NURBS. The answer is *trimming*.

Trimming lets you visually cut or divide a surface along a curve-on-surface so it appears to have holes or missing pieces. The trimmed surface, however, is not actually cut. It exists in a hidden form that does not render or affect modeling. You can recover the trimmed part of a surface using the **Untrim** tool.

Creating curves-on-surface and then trimming is the most common way to combine NURBS surfaces in industrial design.

Shells

Shells are a special type of surface or collection of surfaces you can use for special modeling operations, or for export to solid modeling packages.

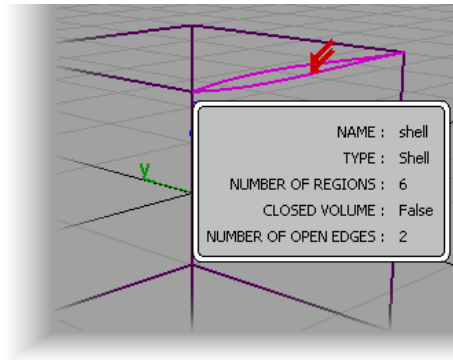
Shells are collections of adjacent NURBS surfaces. Every surface stitched into a shell must meet the edge of another surface in the shell at some point.

Shells are stored as a single node in the DAG.

Shells can be open or closed. For closed shells, the normals should always point outward. This is necessary for the Boolean operations.

The main uses of shells are:

- To improve data transfer to some CAD packages.
Some CAD packages deal with shells much better than normal trimmed NURBS surfaces.
- To prepare for Boolean operations.
The Boolean tools (Shell subtract, Shell intersect, and Shell union) only work on shells. Often you will simply stitch surfaces into shells, apply a boolean operation, then unstitch back into surfaces.
- To check adjacencies between surfaces.
Surfaces can only be stitched into shells if they are within an adjacency tolerance.
If the tolerance is set correctly, you can easily check whether a group of surfaces will export or build properly by checking whether they will stitch together into a shell.
- To identify open edges in stitched shells:
Use **Object edit > Query edit** to check for open edges in shells. Red arrows clearly mark gaps in the shell.



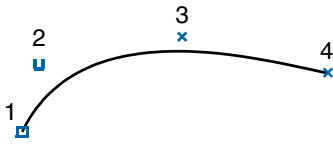
Shells have the following limitations:

- Depending on the options in the Shell stitch option window, a stitched shell may not match the original surfaces exactly.
In this case, unstitching will not produce surfaces that match the originals exactly either.
- You can not edit CVs of a shell. If you need to reshape the surface of a shell, you must unstitch the shell.
- You cannot use the isoparametric curves of shell surfaces as input for other tools.
- You cannot maintain continuity with a shell in tools such as **Square** and **Rail Surface**.
- You cannot create fillet surfaces on shells or between shells and other surfaces.
- If you stitch an object, then scale it, then unstitch it, you not be able to re-stitch the object. This is because the scaling operation can increase the gaps between surfaces, thereby causing any subsequent stitch operations to fail (within the current tolerance settings). In this case, scale the object before you first stitch it.

Object properties

Explains the properties common to NURBS objects.

Degree



Degree is a mathematical property of a curve or of a surface dimension that controls how many CVs are available for modeling.

The number of CVs for each curve span is controlled by the *degree* of the curve. The default curve type in Studio is *degree 3*, which has four CVs for the first curve span.

You can choose to have fewer CVs per span, or, if you have an advanced version of Studio, you can create curves with more than four CVs per span.

- Degree 1 creates curves or surfaces with straight lines.
- Degree 2 curves or surfaces do not automatically have smooth transitions between spans or patches.
- Degree 3 is the default degree for new curves and surfaces.
- Degree 5 and degree 7 curves are generally used in automotive design. They are slower, but give you smoother curves, better internal continuity, and more control.

The degree of your curves can affect data transfer to CAD packages. Some other packages cannot accept curves with degree higher than 3.

Surfaces can have different degrees across their width and length. So, for example, a surface could be degree 3 along its width, and degree 5 along its length.

Parameters and parameterization

Parameters are the unique numeric values (like a coordinate) of points on a curve or surface.

What are parameters?

You can think of a curve as made up of an infinite number of points. Each of these that make up a curve has a number, called its *parameter*. Parameters let you refer to specific points along the length of a curve. The higher the parameter, the further is the point along the curve.

Just as points in space have three dimensions, called X, Y, and Z, the parameters of a point are measured along the one internal dimension (length) of the curve. We call this dimension *U*.

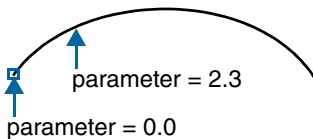
Since surfaces have two internal dimensions (length and width), we need another parameter (in addition to *U*) to specify a point on a surface. We call this parameter *V*.

Just as every point along the length of a curve has a *U* parameter, every point across a surface has a pair of *U* and *V* parameters.

What is parameterization?

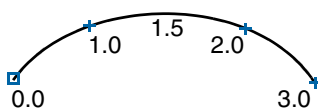
The method Studio uses to number the points along a curve is called the curve's *parameterization*. Studio has two parameterization methods: *uniform* and *chord-length*.

Each method has advantages and disadvantages depending on how the curve will be used. You can choose which parameterization method to use when you create a new curve, and you can rebuild existing curves to use a specific parameterization.

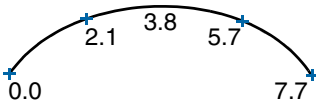


Uniform

Uniform parameterization assigns integral parameter values to the edit points, and evenly distributes parameters along the spans between edit points. So the first edit point is always parameter 0.0, the second edit point is always 1.0, the third is always 2.0, and so on.



A bonus feature of uniform parameterization is that the parameter value of the last edit point is the also the number of spans in the curve. However, unlike chord-length parameterization, the parameters of a uniform curve have nothing to do with the actual length of the curve.



Chord-length

Chord-length parameterization assigns parameter 0.0 to the start of the curve, then increases the parameter value proportionally to the *chord length*, or the shortest linear distance, between the surrounding edit points.

Unlike uniform parameterization, the parameters of a chord-length curve are irregularly spaced between the edit points, and the edit points do not have integral parameters.

Comparison

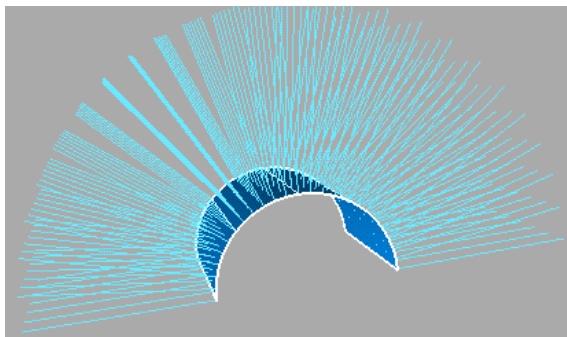
Each parameterization method has advantages and disadvantages, depending on how you will use the curve or surface.

Type	Pros	Cons
Chord-length	<ul style="list-style-type: none"> Parameter value gives some indication of the point's relative position along the curve. Minimizes stretching and squeezing of textures. 	<ul style="list-style-type: none"> Parameters are not obvious. Surfaces built from chord-length curves can be more complex because of cross-knot insertion.
Uniform	<ul style="list-style-type: none"> Easy to reckon parameters (for example, 1.5 is about half-way between edit points at 1.0 and 2.0). 	<ul style="list-style-type: none"> In many cases, interpolation between edit points is not as good. Can lead to unpredictable stretching of textures during rendering.

Just as with degree, surfaces can have different parameterization methods for their U and V dimensions. For example, the U isoparms of a surface can be degree 3 with uniform parameterization, while the V isoparms are degree 1 with chord-length parameterization.

Normals

Normals are imaginary lines perpendicular to each point on a curve or surface.



The direction of U and V isoparms on a surface determines the direction of the surface's normals.

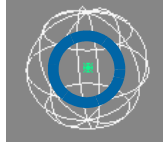
Normals are a mathematical side-effect of NURBS. They are often used as a way of specifying which side of a surface points "inside" or "outside" (for example, when creating *shells*).

Normals are also an indirect indicator of the shape of a curve or surface. Since they are always perpendicular to the curve or surface, the way normal lines point toward or away from each other can reveal subtle curvature.

Pivot points

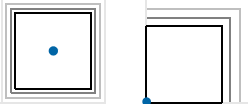
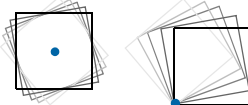
The pivot point is the point around which an object rotates and scales, and which represents the point location of the object when it moves.

When you pick objects in the view windows, you can see a small blue-green dot associated with every object. This is the *pivot point* of the object.



Pivot points allow you to control how objects rotate and scale, and also represent the exact locations of objects in space.

All transformations to an object are relative to the pivot point:

Transformation	Relationship to Pivot
Move	Moves the pivot point (and the object travels along with it).
Scale	Scales object out from or in toward the pivot point. 
Rotation	Rotates object around the pivot point. 

Construction history

Construction history is the saved information about how an object was created. When you edit the construction history the object will automatically update.

For almost every tool, StudioTools gives you the option of saving the history of how an object was constructed. This means you can edit the curves, surfaces, manipulators, tool options, and so on that were used to create an object, and the object will automatically update.

For example, when you use the **Revolve** tool to create an object with construction history, you can:

- reshape and edit the curves you revolved...
- re-display the construction manipulator that created the revolved surface...

...and the surface(s) will update automatically.

To create construction history when working with tools, turn on the **Create History** option in the option window. This option is on by default in all tools.

Objects that have construction history are drawn in green in the default color scheme.



If a surface or curve has been built with construction history, it cannot be moved, scaled, or rotated even if its constructor objects are transformed along with it.

Modeling concepts

Describes general and StudioTools specific concepts that you will use when modeling.

Absolute and relative addressing

Choose whether translations are made in absolute values, or relative to the object's current placement, rotation, and size.

By default, the system addresses view coordinates in Absolute mode as indicated by the (ABS) notation as part of the move prompt on the information line. While addressing in absolute mode, an object will be moved to the grid position specified, or rotated to the absolute degree value specified for each of the three axes, or scaled based on its original size.

If you want to rotate an object on only one or two axes without affecting the rotational position on the third axis, the current values on the axis you do not want to change must be re-entered.

For example, if an object is currently rotated to 45 degrees on both the x and y axes, and you want to change the rotation on the x axis to 65 degrees, the rotational amounts would be entered as 65, 45, 0 at the prompt line and then press the **Enter** key. Trailing zero values can be omitted, so in this case, 65, 45 followed by pressing the **Enter** key would work as well.

You can switch into relative addressing mode at any time by typing a lower case letter "r" followed by the translation amounts. The notation on the information line will change to (REL) to show that the system is accepting input for relative addressing. When in relative addressing mode, objects are rotated the amount specified for each axis, relative to the object's current rotation.

If you want to change the current rotation on only one or two of the axes without changing the current rotational position on the other axis, the values on the axis you do not want changed must be entered as zero.

For example, if an object is rotated to 45 degrees on both the x and y axes, and you want to rotate the object an additional 4 degrees on the x axis relative to its current position, the rotational amount would be input as 4, 0, 0 followed by pressing the Enter key. The zero values for the y and z axes result in no positional adjustment on these two axes.

Once again, trailing zero values can be omitted. In this case, typing 4 followed by the Enter key at the prompt line achieves

the same result as well, since the relative rotational change for both the y and z axis are null.

To switch back to the absolute addressing mode at any time, enter the lower case letter “a” followed by the translation values.



The addressing mode switch (“a” or “r”) can also be typed, followed by pressing the Enter key, without typing in any values.

Momentary and Continuous buttons

There are two types of tools in StudioTools: momentary and continuous.

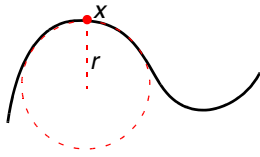
Momentary functions, such as **Pick > Nothing**, perform an operation once, every time you select the function.

A continuous function, like **Pick > Object**, remains selected and highlighted, letting you use the function repeatedly without reselecting the button. To stop a continuous function, just select another continuous function.

You can select momentary function without interrupting a continuous one. When the operation of a momentary function is finished, the system reverts to the operation of the last continuous function. For example, if you select **Pick > Object** and after picking a few things, want to view the scene in another window, select **Layouts > Top**. After the Layouts operation is performed, you will automatically continue in the Pick operation without having to re-select the Pick button.

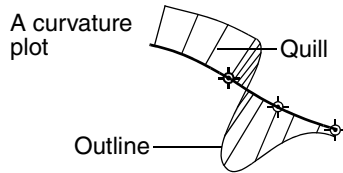
Curvature

Curvature is a measure of how much a curve curves.



Curvature is measured by fitting a circle into the curve, then taking the reciprocal of the circle's radius. In the illustration at left, at point x , the curve is best described by a circle with radius r . At this point, the curvature is $1/r$.

(We use the reciprocal, $1/r$, instead of just r because a flat line has an infinite radius. Taking the reciprocal gives us 0 instead of infinity.)



Several tools in StudioTools, such as the [Locators > Curve curvature](#) tool, allow you to display a *comb plot* of a curve's curvature. At regular points along the curve, the tool samples the curvature, and draws a line (sometimes called a "quill" because it looks like a spine on the back of a porcupine). The length of the line represents the curvature value at that point.

Laying out curves and surfaces

As you create curves and surfaces, or fit curves and surfaces to scan data, you will have to decide how to use separate surfaces to create the overall model.

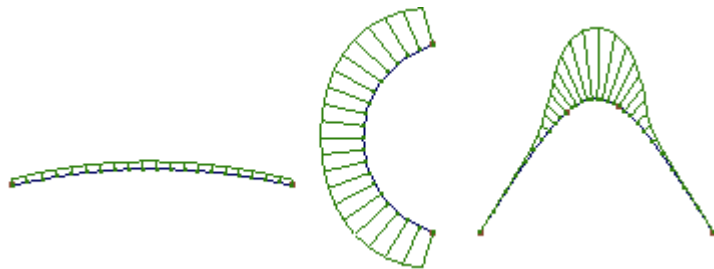
Introduction

For all but the very simplest models, you will not want to create the entire model using a single surface. Sometimes the choice of boundaries between separate surfaces will be obvious. But in cases where there is no clear natural boundary, you will have to decide how to break up a large-scale areas into individual surfaces.

This decision is a bit of an art, with different modelers making different decisions to emphasize different priorities. In this topic, we will attempt to give you a broad overview of the process.

Deciding where to separate surfaces

Consider the following cross sections:



The shape on the left has low curvature. The shape in the middle has high curvature. The shape on the right has two changes in curvature.

You will want to break up large-scale areas into areas of low curvature and high curvature at the points where the curvature begins to increase.

Low curvature

In areas of low curvature, not as many CVs are needed to describe the shape, so you can use a single span and a lower

degree curve. Using separate surfaces for these areas lets you use simpler geometry.

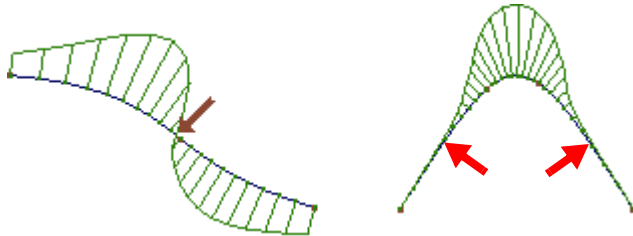
High curvature

In areas of high curvature, you will want more CVs to describe the shape more accurately. Using separate surfaces in these areas lets you use high degree surfaces or multiple spans to get more CVs.

Note that even if you can “get away with” describing the shape with a small number of CVs, the CVs may be doing too much work. That is, each CV is responsible for controlling such a large area of the curve or surface that making small changes to the curve or surface later will be very difficult.

Changes in curvature

You will want to break up shapes where the curvature changes direction (called inflections, shown below on the left), and where curvature begins to change (shown below on the right).



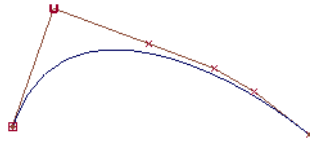
CV distribution

In each case, breaking the model up involves maximizing the use of CVs. That means creating conditions where no CVs are “overworked” (having too much influence on the shape of the curve or surface), and the CVs have a smooth distribution, both of which make maintaining shape and continuity easier.

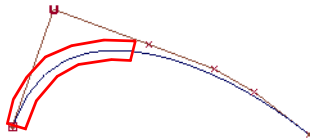
Overworked CVs

Overworked (or high tension) CVs are CVs that are distant from the curve they control, or have a significant influence on the shape of their curve or surface.

In the following simplified example, the second CV in the curve on the left is clearly doing a lot of work: it's almost solely responsible for pulling the shape of the curve to the left.

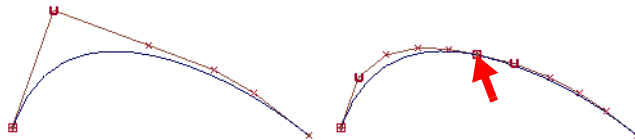


This makes editing the shape of the curve difficult. Because a single CV is largely responsible for the shape of a section of the curve (marked below), and any reshaping you want to do anywhere within section must be accomplished by moving that one CV.



This leads to extremely minute and frustrating adjustments of the CV, as you find each movement affects a larger area than just the small part of the curve you wanted to improve.

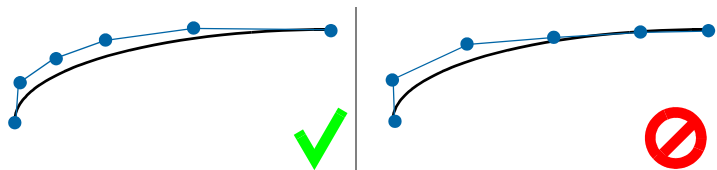
Using separate curves (as shown below on the right) immediately improves the situation. Now each CV in both curves is exerting roughly the same amount of influence.



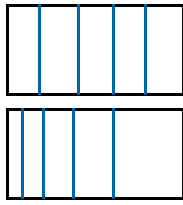
Good vs. poor CV distribution

A good distribution:

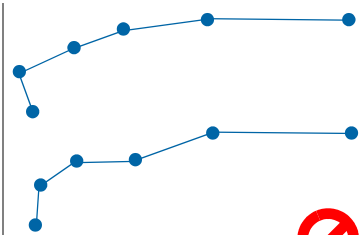
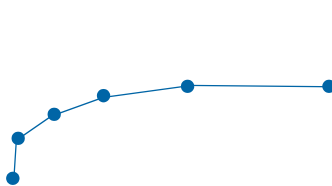
- ◆ Puts more CVs in areas of high curvature.



- ◆ Has even or smoothly changing spacing between hulls.



- ◆ Has consistent direction change along hulls, with no zigzags, W shapes, or pronounced peaks.



Continuity

Continuity is a measure of how well two curves or surfaces “flow” into each other.

Palette tool [Curves > Blend curve toolbox](#)

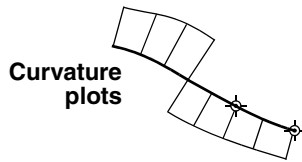
Why you would set continuity and curve degree

- To get more visual smoothness at intersections, increase the level of continuity.
- To increase the amount of flexibility available to achieve high levels of continuity, increase the curve degree.

Types of continuity

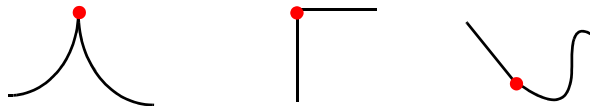
Continuity is a mathematical indication of the smoothness of the flow between two curves or surfaces.

The following lists the five types of continuity possible with StudioTools tools, G0 to G4. Note that G3 and G4 continuity are only available with blend curves.



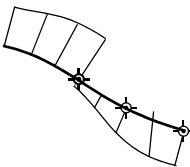
Positional (G0)

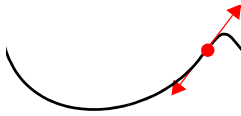
The endpoints of the two curves meet exactly. Note that two curves that meet at any angle can still have positional continuity.



Tangent (G1)

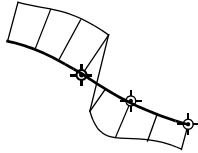
Same as positional continuity, plus the end tangents match at the common endpoint. The two curves will appear to be travelling in the same direction at the join, but they may still have very different apparent “speeds” (rate of change of the direction, also called *curvature*).





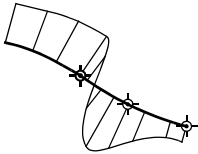
For example, in the illustration at left, the two curves have the same tangent (the double-arrow line) at the join (the dot). But the curve to the left of the join has a slow (low) curvature at the join, while the curve to the right of the join has a fast (high) curvature at the join.

Curvature (G2)



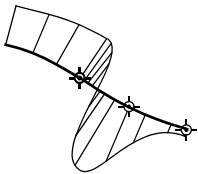
Same as tangent continuity, plus the curvature of the two curves matches at the common endpoint. The two curves appear to have the same “speed” at the join.

Curvature with constant rate of change (G3)



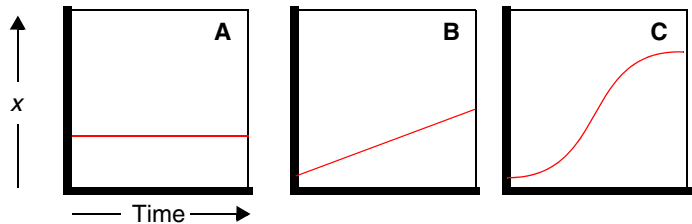
Same as curvature (G2) continuity, plus the *rate of change* in the curvature matches between the curves.

Curvature with constant rate of change of the rate of change of the curvature (G4)



Same as G3 continuity, plus the rate of change of the rate of change of the curvature matches between the curves. This is the smoothest type of join.

The concept of “rate of change of the rate of change” may be hard to conceptualize. Consider the following graphs:



- In graph A on the left, the value of x does not change, so the *rate of change* of x is 0.
- In graph B in the middle, x has a constant rate of change, which we can calculate as the slope of the line.

- In graph C on the right, the rate of change is not constant: it is slow at first, then fast, then slow again. The rate at which the rate of change *itself* changes is the *rate of change of the rate of change*.

The construction plane

The construction plane defines the a temporary coordinate space that can be moved or rotated away from the absolute world space. Any reference plane can be set as the construction plane.

What is the construction plane?

Tools in StudioTools place objects in an XYZ coordinate system. Normally this is the *world space* coordinate system, the absolute frame of reference for your scene.

However, there will be times when you want to align objects where the orientation, position and rotation are different from the world space axes and origin.

A construction plane let you create and work in an alternative coordinate system. When the construction plane is active, the points you click or coordinates you type use the construction plane's coordinate system, instead of world space.

You can position and rotate the construction plane freely, or constrain it in relation to a curve or surface.

You can switch between world space and an active construction plane using the [Construction > Tgl Construction Plane](#) tool.

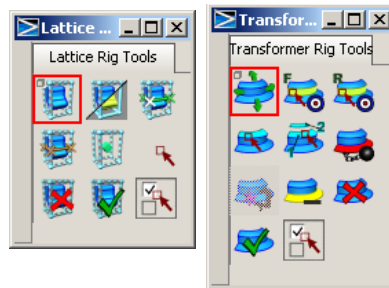
What is the relationship between construction planes and reference planes?

You can have many reference planes in your scene performing various jobs, but only one plane at a time can be the construction plane.

You can tell StudioTools to use any reference plane as the construction plane using the [Construction > Set Construction Plane](#) tool.

Dynamic Shape Modeling

StudioTools has two tools for dynamic shape modeling: a Lattice Rig tool and a Transformer Rig tool.



Providing two separate tools gives you the following benefits:

- The Lattice Rig is an easy-to-use tool that doesn't require deep user experience.
- The advanced Transformer Rig allows a more detailed and specific shaping process.

For information about these toolboxes and tools, see [Object Edit > Dynamic Shape Modeling > Transformer Rig](#) and [Object Edit > Dynamic Shape Modeling > Lattice Rig](#).

Why we developed dynamic shape modeling

The Dynamic Shape Modeling tool family was developed to support global modification of datasets. While it is relatively straightforward to modify just one piece of geometry, it can be a very time-consuming task to change the proportions of an entire model composed of many pieces of geometry. As a designer, you need to explore the proportions of the model. To play with it, you need to be able to modify the whole set of geometry, sometimes as a single unit. The result doesn't need to be a production model, just a model that holds together, expresses the intent of your design, and enables you to make a choice before you finalize your design.

The purpose of dynamic shape modeling

The Dynamic Shape Modeling tools give you the ability to globally change the model easily. Think of it as an advanced non-proportional modification tool that stretches and

compresses the model. The basic relationships of parts of your model to each other will not change, and features can not be added or subtracted, but within the model, relative sizes, proportions, and shapes can be modified.

What Dynamic Shape Modeling doesn't do

There is no guarantee that surface continuities are maintained while the global shape is being modified. After the shape modification, you'll need to check the model, and perform additional work to fix the continuity breaks, as necessary.

What to expect from the tools

Using Dynamic Shape Modeling for communication and concept development

- Use these tools for balancing proportions of geometry sets. The output of the tool may not necessarily provide production surfaces, even when the input is of production quality. This tool can easily be used as a communication tool for designers and surface modelers.

Using Dynamic Shape Modeling for further modeling and modification

- Because the warping of the global shape will not destroy the surface parameterization, the modified surface set can be used for further modeling. This tool can be used for Class A surfacing work, as long as you realize that further work may be required to bring the model back to production quality.

Common concepts of Dynamic Shape Modeling

While the tool sets have significant differences, they share the following concepts.

Targets:

A target is any geometry that is being modified.

Targets can be surfaces, meshes, or curves.

Modifiers:

A modifier is geometry used to define the desired changes to the targets.

Constraints:

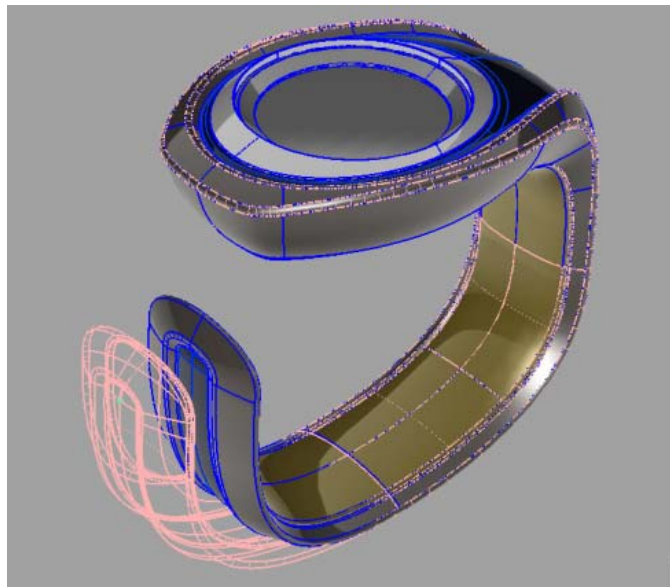
Constraints are used to secure parts of the target geometry to prevent shape modifications.

What happens with the target geometry?

As soon as the targets are picked and accepted, the tools duplicate the targets. Shape modifications will happen on the duplicated model. The original geometry is made invisible.

While you are modifying the shape of the geometry, the dynamic shape modeling tools allow you to toggle the visibility of the originals and the modified geometry. This makes it easier to compare your modifications with the original.

If, for any reason, the duplicated geometry (visible geometry with dynamic shape history) is modified in such a way that the construction history has to be deleted, the original geometry will be set visible and made into a template.



The tools allow you to revert a modification by deleting the duplicated geometry and restoring the original, or commit to a modification by deleting the original and the history.

About the Lattice Rig

The Lattice Rig tool uses a lattice to effect global shape modification. The geometry is shaped by moving the lattice points, which squeeze and pull the model.

Glossary of terms

Target:

The geometry that can be globally modified is called a target. Targets can be surfaces, meshes, or curves.

Lattice:

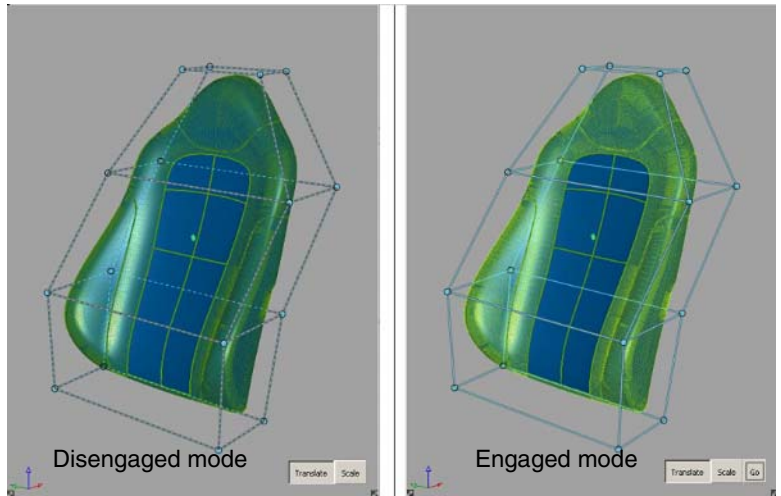
A manipulator used to articulate the desired changes to the targets.

What is the lattice?

The lattice is a manipulator in the Lattice Rig tool. It is initialized as a bounding box around the target geometry.

The lattice has two modes: disengaged and engaged. A disengaged lattice is drawn with dashed lines, and when modified, has no influence on the target geometry. This gives you the ability to refine the lattice to suit the intended modifications of the target geometry.

An engaged lattice is drawn with solid lines, and when manipulated, will modify the target geometry.

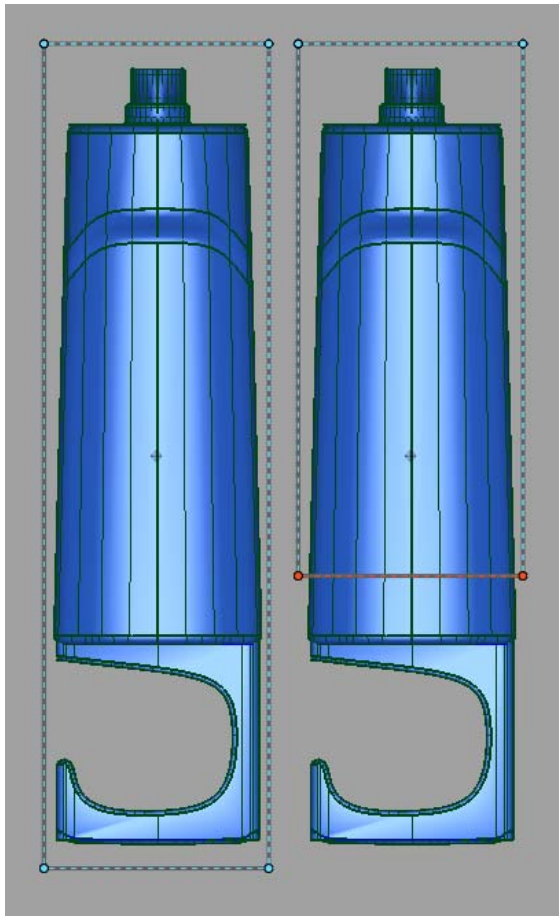


Constraining

You can shape the disengaged lattice so that it does not fully enclose the targets.

When the lattice is engaged, everything that is outside the lattice will remain unchanged; everything that is inside the lattice will be modified.

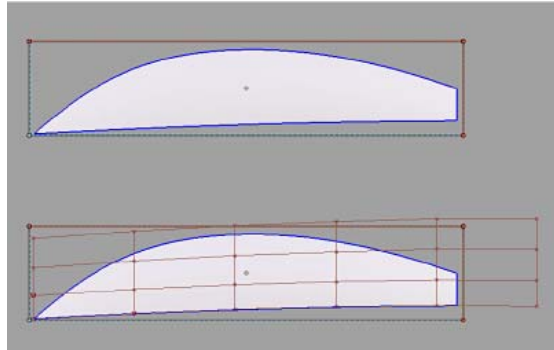
To facilitate this, parts of the lattice that intersect the targets are locked and drawn in red.



In disengaged mode, the lattice points are modifiable; in engaged mode these points are *not* modifiable.



For the purposes of locking the lattice, this tool treats trimmed surfaces as if they were not trimmed.



What is the lattice rig used for?

The Lattice Rig tool is especially suited to general volumetric shape changes to a model. Consider using the Lattice Rig for more conceptual and fast explorations.

About the Transformer Rig

Glossary of terms

Target:

The geometry that can be globally modified is called a target. Targets can be surfaces, meshes, or curves.

Modifier:

Modifiers are geometry used to articulate the desired changes to the targets. In the Transformer Rig toolbox, these are Modifiers.

Constraints:

To secure parts of the target geometry to prevent shape modifications, use constraints. Dynamic shape modeling analyzes the transformer rig and assumes the constraints delimit a *region of interest* (ROI). Parts of the targets that lie outside of the ROI will be clamped (they will not move).

The estimate of the region of interest, however, is not always correct. If there are inaccuracies in the ROI, use clampers.

Clampers

Clampers are hints you place outside the intended region of interest to help the tool understand the desired ROI. Clampers help ensure that modifications don't happen outside the ROI.

Rigid targets

In some cases, there are geometries or objects within the target geometry that should keep their shape. Usually these are parts like buttons, door handles and lights. These rigid targets will be moved embedded in the flexible targets, but they will not lose their shape during the warp. Making a target rigid helps preserve the shapes of the parts while allowing them to move with the surface. Imagine grommets moving on a rubber tarp that is stretched to cover a load: the grommets remain the same shape and size on the flexible surface of the tarp.

What is the Transformer Rig used for?

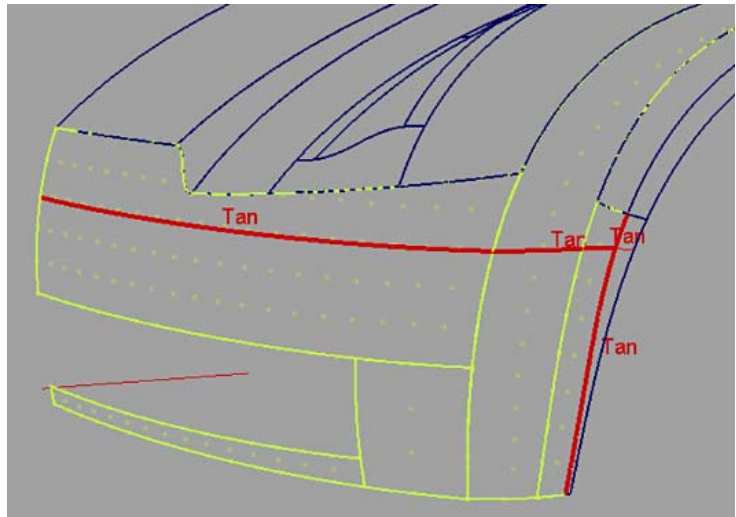
The Transformer Rig tool is used for more controlled shape modifications driven by specific features of the model.

Why choose the Transformer Rig over the Lattice Rig?

Custom modifiers and constraints:

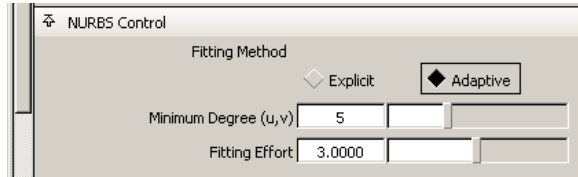
The Transformer Rig enables you to create custom modifiers specific to the model being changed. This provides tighter control of the surface modification.

The Transformer Rig also enables you to constrain parts of the selected target geometry. You can select real geometry to constrain the modifications, which makes the entire warp result more precise, and enables you to make finer-grained changes.



The Transformer Rig offers more flexibility with NURBS fitting options

The Transformer Rig offers an additional NURBS fitting method called *Adaptive*.



See *Set up a Transformer Rig* on page 368.

See *Use Transformer Rigs* on page 370.

See *Change Transformer Rigs* on page 371.

See *Add a clamp to surfaces in Transformer Rigs* on page 372.

See *Use predefined modifiers with Transformer Rigs* on page 373.

See *Use rigid targets with Transformer Rigs* on page 374.

See *Set up a Lattice Rig to modify shapes* on page 376.

See *Use a lattice to modify shapes* on page 379.

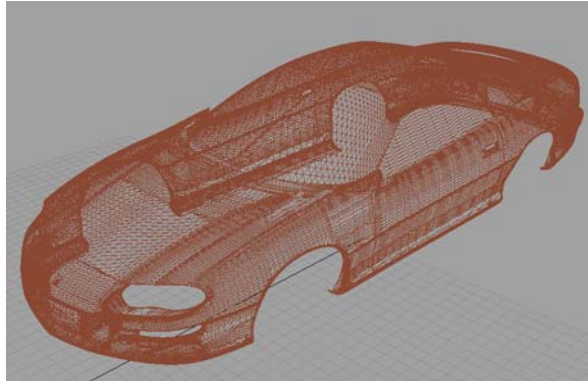
Meshes

Large polygonal objects usually resulting from the scanning and digitizing of physical models.

What is a mesh?

A mesh is a large polygonal object resulting from scanning and digitizing physical objects to create data models in StudioTools.

Meshes can contain several million triangles and, because of their internal representation, are a more efficient way than polysets to store large and detailed data models representing real objects.



Difference between meshes and polysets

How to differentiate between the two types of polygonal objects.

You cannot select and manipulate individual vertices or triangles in meshes like you can in polysets. However, meshes can be separated into pieces that can be shaded independently, deleted, or toggled invisible.

Conversion tools are provided to transform meshes into polysets and vice-versa.

Visual cues


Beside having a different internal representation, meshes are different from polysets in the following ways:

- The default inactive color of a mesh is the same as that of hulls and edit points (brown).
- Using **Object edit > Query edit** on a mesh will show the TYPE as Mesh.

Allowed operations

The following tools can be used on mesh objects:

- All tools from the **Mesh** tab:
- Some **Evaluate** tools:
 - ◆ **Evaluate > Parting line**
 - ◆ **Evaluate > Deviation map**
 - ◆ **Evaluate > Cross section**
 - ◆ **Evaluate > Dynamic Section**
 - ◆ **xsect** tool on Control Panel
- I/O tools from the **File** menu:
 - ◆ **File > Open**
 - ◆ **File > Import > File**
 - ◆ **File > Save**
 - ◆ **File > Save as**
 - ◆ **File > Export > STL**
- Some **Display** tools:
 - ◆ **DisplayToggles > Hardware Shade**
 - ◆ **ObjectDisplay > Visible**

- ◆ **ObjectDisplay > Hide Unselected**
- ◆ **ObjectDisplay > Template**
- ◆ **Diagnostic Shading** modes on Control Panel
- ◆ **Mesh Display** parameters on Control Panel
- Some **Geometry** tools:
 - ◆ **Curves > Blend curve toolbox**
- 

Blend curve tools that depend on parameter values or those related to curvature will not work on meshes since meshes are polygonal objects.
- Some **Locators** tools:
 - ◆ **Locators > Move locator**
 - ◆ **Locators > Annotate**
 - ◆ **Locators > Measure > Distance**
 - ◆ **Locators > Measure > Angle**
 - ◆ **Locators > Deviation > Closest Point**
 - ◆ **Locators > Deviation > MinMax Mesh-Surface deviation**
- Tools that operate on dag nodes:
 - ◆ **Pick > Object**
 - ◆ **Delete > Delete active**
 - ◆ **Transform > Move, Transform > Rotate, Transform > Scale**, etc

See Create and manipulate meshes for more information.

See *Create a mesh* on page 434.

See also Visualize the deviation between meshes and NURBS surfaces.

See *Visualize the deviation between mesh-surface, surface-surface or mesh-mesh* on page 409.

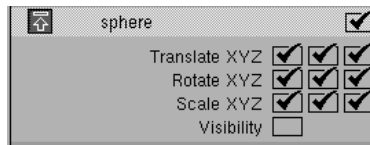
Introduction

Learn what animation is, what you can animate and about the different animation techniques used.

What is animation?

Learn how animation is defined and used in StudioTools.

Animating an object means that one or more characteristics or *attributes* of the object change over time. For example, if you have designed a car and want to see it drive down a road, you must animate its position over time. At time 1, the car may be in front of a house, and at time 50, at a street corner, 10 blocks down the street. In the animation system, you might say that at time 1 the car has an X translation of 0 units, and at time 50 it has an X translation of 10 units (that is, it has moved to a position of 10 units in the X direction). The *X translation*, in this example, is an attribute of the car that can be animated: we call it an animation parameter.



There are a number of ways to animate in StudioTools. Many of our users use the animation tools to present final concept models. For example, you can show the assembly of your model or you can display your new model moving through a scene. The animation and photorealistic rendering capabilities create images convincing enough to be reproduced directly into print, video or interactive media.

The animation process is to model, animate, fine tune, and to finally render your animated scene.

What can you animate?

Learn about different levels of animation available in StudioTools.

An object generally has many attributes, or animation parameters, that can be animated. In StudioTools, a directed acyclic graph *node* has ten attributes that can be animated: the X, Y, and Z translation, rotation and scale attributes, and also the visibility.

Other types of objects have different animation parameters. For example, a camera's angle of view can be animated, and a light can have its color or intensity animated. An object or other item that has at least one animation parameter or attribute that can be animated is called an *animatable item*.

StudioTools offers three different levels of animation

- Keyframe animation
 - ◆ create a turntable animation
 - ◆ create a motionpath animation
 - ◆ add a camera to your animation
 - ◆ create an exploded assembly view animation
- Skeleton and Inverse Kinematics
 - ◆ create skeleton models
 - ◆ apply IK handles to move parts of the skeleton
 - ◆ use mathematical expressions to determine **action** and motion
- Advanced Animation
 - ◆ animate and deform surfaces
 - ◆ deform time with time warps

Basic workflow for manually creating an animation

Learn the process of modeling, animating, fine tuning, and finally rendering your animated scene.

StudioTools provides two types of automatic animation, where you plug in parameters and StudioTools creates the animation, as well as manual, freeform animation.

In StudioTools, manually creating animation involves establishing a *timeline*, then varying one or more properties of objects (for example, position or color) over time.

To apply the workflow

- 1 Create the model.
- 2 Decide how long you want the animation to be and create the necessary number of time frames in StudioTools.
- 3 Use basic techniques to vary the scene through the length of the animation:

- ◆ Place objects you want to animate, including the camera, where you want them, and with the values you want, at each point in the timeline, then mark those frames as keyframes.

or

- ◆ Establish *motion paths* for objects to move along through time.

For more advanced animation, StudioTools is capable of varying almost every property of an object or shader along the timeline, not just position.

- 4 Decide how the objects should transition from frame to frame.

More advanced animation can use the *Action window*, *expressions* (mathematical formulas describing relationships between time and object properties), and *constraints*, to create more realistic and automated effects.

- 5 Preview or render the animation.

Parameters

Objects have many parameters that can be animated. Examples are the objects X,Y, and Z positions, rotations, scaling, and visibility.

Different types of objects have different animation parameters. For example, you can animate a camera's field of view, and the color and intensity of the light.

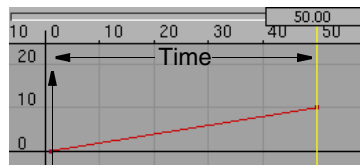
In StudioTools, you control which parameters of an object are animated using the Param Control window.

What happens when an item is animated?

Learn how a channel describes how its animation parameter can change values over time.

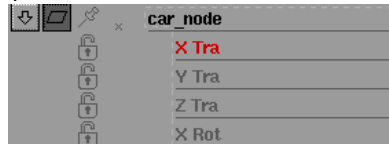
When an animation parameter of an item is animated, a *channel* is created which belongs solely to that animation parameter. The channel describes how its animation parameter changes values over time. When you view a model at different times, the channel is responsible for telling the animation parameter that it now has a different value.

The red diagonal line on the graph shows the animation of the car.

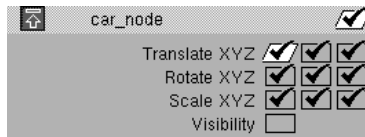


Distance travelled

Translate X is an animated parameter. It's red.

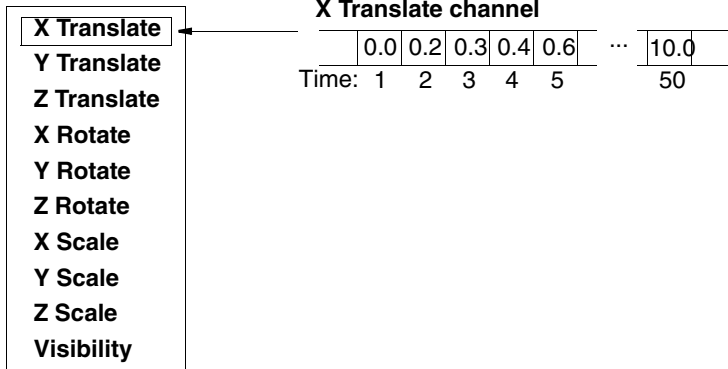


In the Parameter Control window, the **Translate X** channel is a white slanted box. This means that Translate X is animated.



To illustrate: in the car example above, at time 1 the channel tells the animation parameter it has a value of 0. At time 50, the channel tells the animation parameter to assume a value of 10.

Animation parameters of a DAG node:



An animatable object

An object is animated if at least one of its animation parameters has a channel. In StudioTools, a channel is created for an animation parameter by using one of the many animation tools, such as **Animation > Keyframe > Set keyframe**. If you later decide to remove the animation, you can use **Delete > Animation > Delete channels** to remove the channel of animation.

How does the channel know which values the animation parameter should assume at different times?

Parameter curve actions



In the simplest case, a channel evaluates a two-dimensional curve, which plots time against value. These two-dimensional curves are called *parameter curve actions*. The channel tells the action at what time to evaluate, and the action produces an evaluation value.

Example

Actions are often created when a channel is created. What the action looks like depends on the animation tool that was used

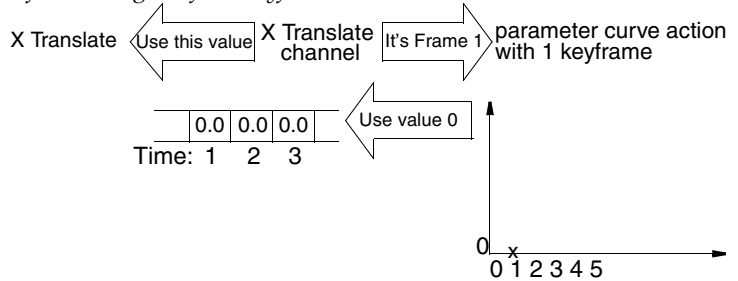
to create the channel. Using the car example, you can describe the car's animation using **Animation > Keyframe > Set keyframe**. When you begin, the **X Translate** animation parameter has no associated channel.

Position the car at 0 units on the X-axis, and set a keyframe at time 1.

Before animating the car:

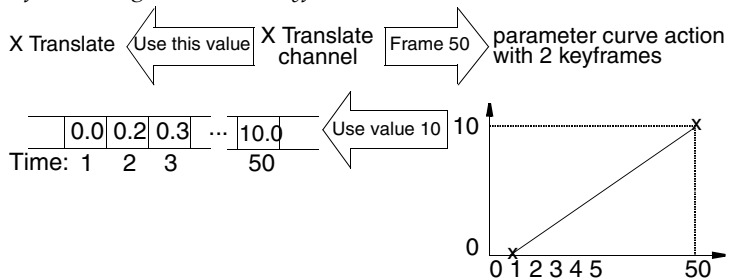
X Translate (no channel)

After setting the first keyframe:



Since the **X Translate** animation parameter was not previously animated, a channel is created for it. The channel needs an action to tell it what values to use, and so a parameter curve action is created that has only one keyframe at time 1. Now move the car into position at 10 units on the X-axis, and set a keyframe at time 50.

After setting the second keyframe:



Since the **X Translate** animation parameter is already animated, you do not have to create its channel. The channel tells the action to insert a second keyframe at time 50. The action is now a curve defined between the times 1 and 50.

How can I tell if something is animated?

Learn how to quickly identify if something is animated in your model.

To see whether an animation parameter has a channel, look at the **LOCAL** parameters for the animated item in the parameter control window (**Animation > Editors > Param control**). An animated parameter has a white slanted box next to its name.

What is a parameter curve action and a motion path action?

Learn more about actions and timewarps.

There are two types of **actions** in StudioTools: *parameter curve action* and *motion path action*. The parameter curve action is a two-dimensional plot of time versus value. A *motion path action* is simply a reference to a 3D NURBS curve. It is evaluated in the following way: the channel gives a percentage value to the motion path action. The motion path action uses this percentage to determine the 3D point that corresponds to that percentage along the curve. This 3D coordinate (X, Y, Z) is returned to the channel. The channel then extracts one of these components (X, Y, or Z), and uses this value as the value for the channel.

Usually a channel is not animated by a single motion path action, but also has a parameter curve action to specify the animation's timing along the motion path action (**Animation > Tools > Set motion**; see *Create a motion path animation* (page 54)). In this case, there is no longer a simple relationship of one channel to one action. The channel uses two actions to determine what values to tell its animation parameter to assume.

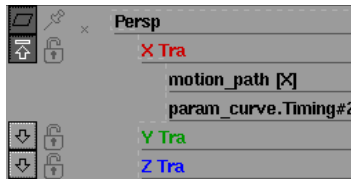
In both these cases, the channel is animated by a *base action*, and each additional action applied to the channel is called a *time warp*. This ability for a channel to use many actions is called a one-to-many relationship, because one channel uses many actions to determine what values its animation parameter should assume.

What is a many-to-one relationship?

Having many channels use the same action is a many-to-one relationship. Since there is both a one-to-many (for example, one channel using several actions) and a many-to-one (for example, several channels using one action) relationship between channels and actions, the combined relationship is actually many-to-many. That is, any number of actions can be associated with any number of channels. The many-to-many relationship between channels and actions provides a greater degree of flexibility in creating your animations.

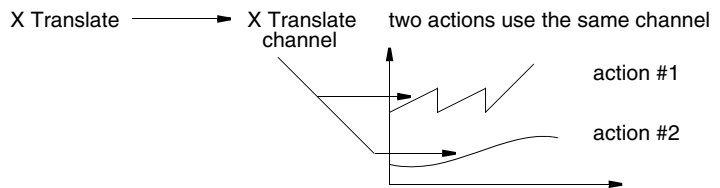


If a channel uses more than one action, then the channel has an expand channel button next to its name in the **Action Window**. If you press this button, you see the list of actions that a channel uses.

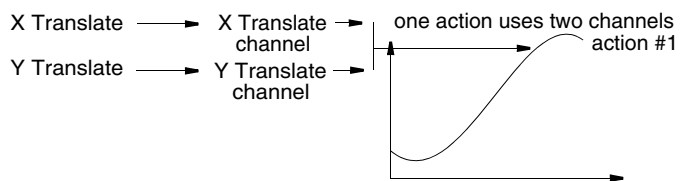


To see which channels use an action, you can select the action and choose **Curve Tools > Show instance** in the **Action Window**.

One-to-many channel to action relationship



Many-to-one channel to action relationship



Summary

The three concepts in the animation system are:

- an *animation parameter* is an attribute of an item that can be animated.
- a *channel* is a set of data that describes what values its animation parameter should assume at different frame times.
- an *action* is a mapping of value versus time.

In the **Action Window**, the relationship between an animation parameter and its channel is made implicitly by using the same name for both.

Example

The animation parameter named **X Translate** is animated by a channel named **X Translate**. In the **Action Window**, if an animation parameter is not animated by a channel, the animation parameter name is listed in light grey. If on the other hand, the animation parameter is animated by a channel, then the animation parameter is listed in red (X parameters), green (Y parameters), blue (Z parameters), or black (all others).

See [Animation > Editors > Action window](#), **Edit > Edit expression** for information on *Expressions*.

Instead of using actions, an animation parameter may be animated by an expression channel. Expressions can be entered by double-clicking next to the animation parameter name in the **Timeview Window**, or by selecting an animation parameter and choosing **Edit > Edit expression** from the Action Window.

What happens when you animate a camera on a curve?

Learn more about the three camera components: camera eye, camera view, and camera up vector.



Geometry in a scene cannot be animated using the [Animation > Tools > Autofly](#) function.

The camera is animated to travel along any NURBS curve.

Camera eye

The *camera eye* can be considered the camera unit itself, which travels along the *motion path* to which it is assigned. The motion path determines the position of the camera at any given time in relation to the scene. For example, the design of the motion path lets you move the camera closer to or further away from an object in the scene. To visualize, think of yourself walking along with a camera taking pictures: the route you follow would be the motion path.

Camera view

The *camera view* can be considered the focus point of the camera — where the camera is looking at any given time. If the camera view is not assigned to a motion path of its own, then the view is always directly in front of the camera. Think of yourself walking through a scene without ever pivoting your head, your view is always directly ahead of your body. By assigning the *view* to a motion path of its own, you can change the view point of the camera at any time in relation to the camera position.

Camera up vector

The *camera up vector* can be considered the current angle of the camera at any given time in relation to the camera eye. The *camera up vector* is the direction from the camera's *eye* to the camera's *up*. If the camera's *up* is not assigned a motion path of its own, the camera remains parallel to the path that the eye has been assigned to at all times. By assigning the *camera up* to an independent motion path, the camera can be pivoted to any angle up to 360 degrees. This can be likened to a camera on a tripod except that the tripod could pivot a full 360 degrees.

At least one curve is needed to use Autofly. If only one curve is used, it must be the motion path for the camera eye. The camera view then remains directly in front of the camera throughout the animation. This might be exactly what you want. If not, and you want to control the camera view throughout the animation, you need at least two motion path curves.

Can I reuse animation on another channel?

Learn how actions are reuseable.

The **Action Window** has tools to make an action reusable by more than one channel. This means that the actions are not *owned* by a channel. They can be renamed to any name, independent of any channel with which they may currently be associated. Actions can also be shared by more than one channel (**Curve Tools > Paste Instance** in the **Action Window**).

In the example above, imagine you want to use the car's motion for a bicycle going down the road next to the car. You can use **Curve Tools > Paste instance** in the **Action Window** to associate the same action used by the **X Translate** channel for the car to be used by the **X Translate** channel of the bicycle as well. Now the car and the bicycle animate together. The advantage of using the same action for both channels is that if you edit the action, then the motion for both vehicles changes.

Example

If you want the car and the bicycle to stop at a house along the way at time 20, and then start moving to the street corner at time 30, you can add two keyframes to the single action, and the animation is modified for both bicycle and car.

What is inverse kinematics?

Learn more about the process and workflow of IK animation.

You can create the illusion of life by using the skeleton and inverse kinematics animation tools in StudioTools. The essence of character animation is timing and motion.

Build a skeleton by creating and editing joints and bones. After you've created all the joints and bones that make up a skeleton for your character, you'll want to move the skeleton around and put it in various poses.

There are two basic ways to pose a joint chain: forward kinematics and inverse kinematics.

With forward kinematics, when you pose a joint chain you have to specify the rotations of each joint individually, starting from the parent joint on down to all the joints below.

With inverse kinematics, when you pose a joint chain all you have to do is tell the lowest joint chain's hierarchy where you want it to be, and all the joints above it will rotate automatically. Inverse kinematics offers a very intuitive way to pose a joint chain because it enables goal-directed posing. When you reach for an object, you don't think about how you are going to rotate your shoulder, your elbow, and so on. You just think about where the object is that you want to reach, and your body automatically does the rest. That's how inverse kinematics works, too.

To pose a joint chain with inverse kinematics you need to add some special tools to a skeleton. These tools are called inverse kinematics (IK) handles. An IK handle enables you to pose a joint chain intuitively.

An IK handle begins at a joint chain's parent joint and can end at any joint below the parent joint. For example, for each leg you could create an IK handle that controls the joint chain beginning at the hip joint and ending at the ankle joint.

You can select the IK handle where it ends at the ankle joint and move the chain with it in the same way that you would think about moving your own ankle.

In addition to posing a skeleton, IK handles also play an important role in the animation of the skeleton. The movement

of a chain between the keyframes of an animation is also automatically solved by the chains IK handles.

IK handles figures out how to rotate and move all the joints in the chain for you by applying an inverse kinematics solver. The IK solver is the motor intelligence behind and IK handle.

You can animate a skeleton, but such an animation would show only the timing and motion of a character lacking form and shape. The next step is to bind the character's model to the character's skeleton so that the skeleton can control the model's actions.

Use **Animation > IK > New skeleton** to create the skeleton, then **Animation > IK > Add IK handle**, **Animation > Tools > Create constraint**, and **Animation > Keyframe > Set keyframe** or **Animation > Keyframe > Auto keyframe** to animate your character in its rotation scale and translation parameters.

Create hierarchical geometry for the character independently from the animation and use **Animation > Editors > Skeletons** to turn DAG nodes in the hierarchy into joint DAG nodes. Then use **Animation > Edit > Overlay skeleton** to overlay the corresponding joint nodes in the model.

What is a time warp curve?

Learn how time warp curves work.

A time warp curve is an animation curve (or a action) that is applied to a channel (the animation of a parameter of an item), and modifies the times at which the other actions in the channel are evaluated.

A channel of an item is usually animated by one animation curve, the base action of the channel. If a time warp curve is applied to the channel, then the channel is now animated by two actions. The time warp action modifies how the base action of the channel is evaluated.

You can apply any number of time warp curves to a channel, and each successive time warp curve modifies the timing of the curve directly below it.

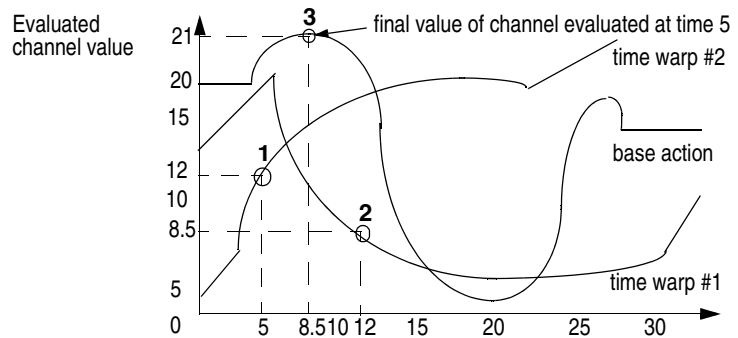
How does a time warp work

The time warp curve changes the timing of a channel by mapping the old time of an animation on the Y-axis to the new absolute time on the X-axis. Another way to look at this is that at a given time on the X-axis, the time warp curve is evaluated to a value on the Y-axis, which is a new time. The time is used as the time at which to evaluate the next action in the channel to which the time warp is applied.

Example of a time warp

As seen in the following diagram, a channel is animated by a base action with the time warp curve time warp #1 applied to the base action and the curve timewarp #2 applied to the curve timewarp #1.

In this example, the channel is evaluated at the time 5.



- 1 First, evaluate the last action in the channel, which is timewarp #2, at time 5. Notice that it evaluates to 12.
- 2 Use the new time and evaluate the next curve in the channel, which is timewarp #1 at time 12, and see that it is evaluates to 8.5.
- 3 Use this new time and evaluate the first and final curves in the channel, which is the base action, and it evaluates to 21. Therefore, the value of the channel at time 5 is 21.

When you create time warps curves, they have a default out-of-range type of identity (from the Action Window' Disp Tools > show infinity menu). This means that before the first keyframe and after the last keyframe of the time warp curve, the time warp curve does not alter or warp the timing of the actions below it.

If many time warp curves are applied to a channel, it is often difficult to fine-tune special areas of the animation in the channel. When you are satisfied with the general animation of a channel with its time warps, you can use **Curve Tools > use result** found in the **Animation > Editors > Action window > Action Graphic Editor** to collapse all the time warps onto the channel's base action. A single parameter curve action is created that evaluates to the same values as the channel with all its time warps. The channel's animation is now replaced by the single resulting parameter curve action.

Rendering

Describes the process of and theory behind rendering images of your 3D models.

The rendering workflow

Describes the basic steps that go into rendering an image.

Rendering your scene in StudioTools uses a similar workflow to photographing a scene in the real world:

- 1** First you must create and lay out the scene, placing objects in position for the shot.
- 2** Next you *assign shaders* to your model. This is somewhat analogous to painting objects in the real world, except that shaders control much more than just the color of the objects. They control what kind of material the object appears to be made of: plastic, chrome, leather, smoke, water, and so on.

If you do not add shaders to the model, objects will render using the default matte blue color.

- 3** Next, you must *light the scene*. Without light, the scene will be completely dark and the camera will simply photograph pitch black.
- 4** Next, you *position the camera* to get the angle you want on the scene. The camera in StudioTools is a near-perfect simulation of a real camera.
- 5** Next you can *preview the shot*. In the real world this might involve looking down the viewfinder. In StudioTools, it means checking the shaded view, and making quick test renders to make sure the scene is set up correctly.
- 6** Finally, you *render the scene*. In the real world, you snap the shutter, rendering the picture on film. In StudioTools, you use the Render command to pass the scene to the renderer, which saves the picture in an image file.

Shaders





Shaders are descriptions of surface materials or effects that control what a surface looks like when it is rendered.

To render a surface, you need a description of what the surface is supposed to look like. Is it gray or red? Smooth or bumpy? Shiny or matte? Should it look like gold or gravel? This description of what a surface should look like is called a *shader*.

Shaders have literally hundreds of parameters that let you simulate virtually any material you can imagine. But this infinite variety is based on two basic decisions: what *shading model* to use, and how to set or apply *textures* to the parameters of that shading model.

Shading models

A shading model is a representation of how light bounces when it hits a surface. Different models simulate different types of materials better.

Model	Useful for...	Example
Lambert	Matte materials (chalk, matte paint, unpolished surfaces).	
Phong	Glassy or glossy materials (plastic, glass, chrome).	
Blinn	Dull metallic materials (brass, aluminum).	
Lightsource	A special lighting model that has <i>no shading</i> . Can be used to represent, for example, the surface of a switched-on lightbulb. Objects shaded with this model do not actually cast light into the scene.	

These shading models and their parameters (for example, color, shininess, and reflectivity) determine the base look for a shader. You can then modify the base look by mapping different types of textures onto the shader parameters.

Textures

Textures allow you to vary the look of a shader across a surface by mapping values to shader properties.

The basic shaders have properties that can be measured at each point on a surface: color, shininess, displacement, and so on. In a new shader these values are uniform, so for example the entire surface has one color.

To create more interesting materials, you can map a *texture* onto the properties of the shader. There are many different types of textures available: color ramps, checkerboard patterns, fractal noise, and more.

For example, you could map a blue-to-green ramp to a shader's *color* parameter, and a checkerboard pattern to a shader's *reflectivity* parameter, to create a material with a smooth transition from blue to green across the surface, and that alternates between reflective and dull in a checkerboard pattern.

Rendering methods

Describes the different methods available in StudioTools for rendering images, and the pros and cons of each.

Raycaster

Raycasting produces smooth shaded renderings that include shadows. Raycasting is faster than raytracing, but does not produce reflections or refraction (although you can simulate these using clever shaders).

Raytracer

Raytracing produces smooth shaded renderings that include optical effects such as reflections and refraction. It is the most realistic rendering possible, but much slower than raycasting.

Hidden Line Renderer

Hidden line rendering produces outline renderings of objects that are filled with flat, unshaded color. Silhouettes of surfaces include the effect of any bump or displacement maps.

Introduction

Learn the theory behind CAD data transfer and how it works within StudioTools.

It is not necessary to read this information to complete a data transfer. However, it may help you understand how data can be transferred successfully.

Introduction to Data Transfer

Explains the general workflow of data transfer used in StudioTools.

StudioTools provides translators of industry standard data exchange formats as well as DirectConnect's file formats.

Data transfer workflow

- 1 Set your model environment to match your CAD system.
Preferences > Construction options > Construction Preset
- 2 Create your model.
- 3 Use evaluation tools to verify that the geometry is suitable for data transfer.

- **Evaluate > Check model**

Use this tool to analyze a model (or portions of a model) for geometry that has particular characteristics. Depending on the options you choose, a report is generated describing the contents of a model and the results of the checks performed.

See *Prepare a model for import into CAD systems* (page 402) for a sample workflow.

- **Evaluate > Continuity > Surface continuity**

This tool checks the position, tangent and curvature continuity between and within surfaces.

- **Locators > Deviation**

Use these tools to check the maximum distance between surface boundaries in StudioTools to confirm the integrity of the model before transferring it to the target CAD system.

- **Surface Edit > Stitch > Shell stitch**

This tool enables you to create a valid solid model topology within StudioTools. Stitching surfaces within StudioTools creates a shell. When the shell is exported to a downstream (CAD) system, it includes an extra layer of information.

The stitching process also identifies surface boundaries that exceed the prescribed tolerances. These problems can then be corrected by the designer prior to the translation of the data.



Save the original model before stitching.



Stitching is not required prior to transferring Unigraphics , Pro/ENGINEER, or CATIA files. If the geometry is stitched, it comes into the target system with topology information. If it is not stitched prior to transfer, the geometry comes into the target system as NURBS geometry.

Learn how Solid Modeling Theory works

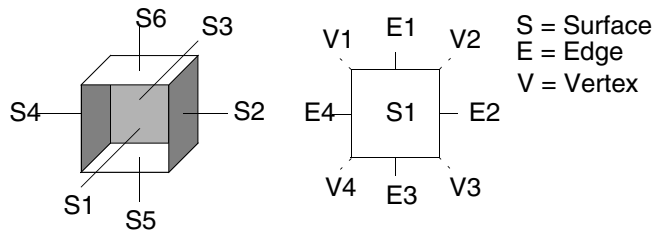
Learn how geometric and topological data works together to form a solid model.

You should be familiar with the concepts of solid model data types to understand how geometric and topological data work together to form a complete model.

- Geometric data
Surfaces contain the geometric data of a solid model. The geometric data describes the basic shape of an object and can be represented using NURBS (Non-Uniform Rational B-Splines).
- Topological data
Loops, edges, and vertices contain the topological relationships between the individual surfaces that form the solid model. Topological data describes how the geometric components are connected together. In solid modeling terminology, surfaces are called *faces*, and each face is made up of loops, edges, and vertices.

How does it work

The following is an example of a solid model of a cube in the which has six surfaces (faces) labeled S1 to S6 which form the geometrical and topological information required to define it as a solid model. Each of the surfaces has a loopset, but in this case each surface has only one loop. The loop for S1 has four edges and four vertices.



The edges are used to connect two loops from adjacent surfaces. The vertices are used to connect two or more edges.

This solid box consists of six surfaces, twelve edges, and eight vertices that form the geometrical and topological information required to define it as a solid model. For example, if a hole is placed in the box through S1 and S6, S1 and S6 would each have a loopset containing two loops.

Learn the Solid Modeling workflow

Learn what requirements are expected in a solid modeler.

Creating solid geometry from imported StudioTools models is a common workflow when integrating StudioTools and CAD systems. When exporting the model from StudioTools, you must ensure that the geometry is built to the correct tolerance and that it can also be stitched. Stitching in StudioTools identifies gaps between surfaces so that you can repair the geometry before exporting the file.

During stitching, the surfaces are twinned. This means that the surface boundaries may be split to accommodate adjacent surfaces, and periodic geometry is detached into multiple surfaces. For this reason, you should save the StudioTools wire file before stitching so that if further modifications to the StudioTools model are required, the construction history will be intact.



A stitched geometry saved to an StudioTools wire file cannot be unstitched to its original state.

What happens when you import data into a solid modeler

When you import a StudioTools model into a solid modeler, you provide the geometric and topological information of the model. When creating a solid model, the solid modeler system creates a valid data base from the supplied data, and the supplied data must satisfy the solid modeler's rules for topological and geometric data.

The topology of a model defines how each surface relates to all other surfaces in the model. The important element of getting the topology right for data exchange to solid modelers is that an edge on one surface must have a “twin” edge on the adjacent surface. Edges are defined by natural surface boundaries or trimmed surface boundaries.



You can transfer surfaces and complete the stitch procedure in a CAD system or first stitch them in StudioTools and then export the data.

Learn about the tolerance requirements for Solid Modeling

Learn how to achieve the tolerances required in solid modeling.

To achieve the tolerance required by solid modeling, it is important to manage the modeling units and tolerances when creating your model. The millimeter (mm) or inch is generally used as the base linear unit. Standards for tolerances have been developed as they apply to engineering-based CAD systems.



If you are not sure of the standards your companies or clients use, ask your CAD system manager. Set up your units and tolerances at the beginning of your modeling session and save them as a preset in the **Construction Options** box. The next time StudioTools is opened, the preset that was in use when StudioTools was last exited, will be in effect.

To successfully join or align surfaces in the target system, the gap between the surfaces of your model must be less than the accuracy defined within the solid modeler.



To specify various tolerances choose **Preferences > Constructions options**.

Rational and non-rational geometry concerns for data transfer

In the **Preferences > Constructions options** window, you can specify whether or not the new geometry being created will contain the rational or non-rational component.

Rational geometry contains CVs that do not have a uniform weight, while the CVs of non-rational geometry all have the same weight. Some CAD systems that do not support rational geometry will rebuild the rational element of geometry upon import. This will change the intended design and therefore the user should know ahead of time whether rational geometry is supported by the target CAD system.

Rational fillets are created with fewer isoparametric curves and the tangency to the adjacent surface can be up to ten times more accurate. While this is an advantage in StudioTools, it is

even more apparent when the geometry has been transferred to a solid modeler. The closer adjacent surfaces are to exact tangency, the more usable the imported StudioTools data is in downstream operations. For example, the further the geometry can be offset during the thickening operation.

Once the above conditions have been met, you should try several sample translations to verify that the geometry is being passed from StudioTools successfully. Before modeling a project in StudioTools that is intended for export, you should model several sample pieces of geometry in mock modeling situations, then transfer them and attempt the stitching operation in the target CAD system. This will confirm that the model, when completed, will transfer successfully.

Whether you are creating a model, verifying a model, or debugging a translation, there are a number of tools in StudioTools you can use to check the quality of the geometry you have created. The most useful tools are the surface continuity checker (**Evaluate > Continuity > Surface continuity**) and the **Min/Max** measurement tools (**Locators > Deviation**). Use these tools to check the maximum distance between surface boundaries in StudioTools to confirm the integrity of the model before transferring it to the target CAD system.

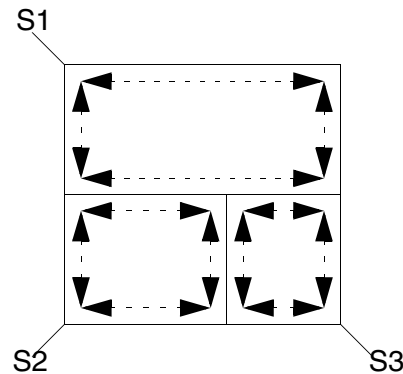
Learn how to get the topology right before transferring data

Learn how to get the topology correct for solid modeling.

The **Surface Edit > Stitch** tools in StudioTools creates a valid solid model topology.

- Stitching surfaces can greatly improve the data transfer to a solid modeler.
- The stitching process also identifies surface boundary gaps that exceed your tolerances.
- The stitching process identifies duplicate surfaces in the model and unifies the direction of the surface normals of the completed shell.

The following illustration shows three individual surfaces (labeled S1, S2 and S3). The edges of each surface are represented by dashed lines.



When models are constructed using the StudioTools advanced surface tools (**Swept**, **Rail Surface** and **Square**), it is common to create a number of smaller surfaces along the edge of one larger surface. This modeling technique does not create the twin edges required for a solid model. Stitching adds this information.



Some modeling techniques, such as **Trim**, **Intersect** and **Round**, create twin edges.

Requirements and workflows for CAD packages

Review the requirements and data transfer workflows.

Pro/ENGINEER

PTC Granite™, IGES or STEP file formats can be used to transfer StudioTools models to or from Pro/ENGINEER Wildfire.

Pro/ENGINEER Requirements

- Pro/ENGINEER Wildfire 1.0 or 2.0, IGES, STEP, or Granite translators.
- SurfaceStudio, AutoStudio, Studio or DesignStudio.

Model Preparation

Units

When working in Pro/ENGINEER set the Units to be the same as what was used in the StudioTools model.

Geometry/Topology

When using StudioTools two types of model information can be sent to and read by Pro/ENGINEER. Those two types are geometry information and topology information.

The Studio-created IGES file includes only the description of the geometry information. The Studio-created STEP and Granite file supports both the geometry information as well as the topology information.

The geometric data describes the basic shape of the object and in both StudioTools and Pro/ENGINEER, geometric data is represented using NURBS.

Topological data describes how the geometric components are connected together to form a solid. The StudioTools STEP file format has advantages over IGES when transferring StudioTools models to Pro/ENGINEER because there is more information describing the model that is being transferred.

Tolerances

- In the Construction Settings window:
Set **Preferences > Construction options - Construction Preset** to **Pro/ENGINEER**.

Information specific to IGES

- StudioTools sets and IGES levels.

StudioTools set information is only exported in files for IGES, VDAIS, or JAMA-IS, if the option Level Mapping is set to SET. If a StudioTools set is given a name of the form of LEVEL<n>, where <n> is an IGES level number (and greater than 0), then the corresponding IGES entity for each member of the StudioTools set is assigned to level<n> in the IGES file. For example, the IGES entities corresponding to each member of the set LEVEL245 is assigned to level 245 in the IGES file.

If a StudioTools object is a member of several multisets that conform to this naming convention, then the IGES file contains a Property Entity 406 form 1 (Definition Levels) listing the IGES levels to which the corresponding entity belongs.

Information specific to Granite



The translation time of rational geometry (for example, exact radius surfaces) is longer than the translation of non-rational geometry. In StudioTools you can create non-rational surfaces and translate them into PTC Granite.



Curve Fit Distance is the key to translation quality. The recommended tolerance in the **Preferences > Construction options - Construction Preset - Tolerances > Fitting** is based on testing done translating models between StudioTools and Pro/ENGINEER.



Maximum Gap Distance is the value that is used to check if the adjacent boundaries are built closely enough to each other. They should never be smaller than the Curve Fit Distance.

Workflow

Before transferring geometry between StudioTools and Pro/ENGINEER, you should consider the purpose of the

transfer to plan an appropriate workflow. When you import your StudioTools model into Pro/ENGINEER, you provide the geometric and topological information of the model. When creating a solid model, the Pro/ENGINEER system must create a valid Pro/ENGINEER data base from the StudioTools supplied data. The StudioTools supplied data must satisfy the Pro/ENGINEER's rules for topological and geometric data.

Get the geometry right

To achieve the tolerance required by solid modeling it's important to manage the modeling units and tolerances when creating your model. Most engineering organizations use the millimeter or inches units as the base linear unit and have developed standards for tolerances that they apply to their CAD systems.

If you are not clear as to which standards your companies or clients use, seek out your CAD system manager. Set up your units and tolerances at the beginning of your modeling session and save them in your `usr_options` file.



The maximum distance or gap between the surfaces of your model must be less than the accuracy defined within Pro/ENGINEER for successful joining of surfaces. The Pro/ENGINEER system defines accuracy as a value less than the ratio of the length of the smallest edge of a part divided by the length of the largest side of a part.

You can lower the part accuracy to successfully join surfaces when the gap exceeds the required tolerance. However, we recommend that your StudioTools models are constructed to within the accuracy defined by the engineering requirements of your organization.

A recommended tolerance to maintain during transfer from StudioTools to Pro/ENGINEER is dependent on the size of the part being described. StudioTools uses an absolute tolerance system to describe geometry which means that every piece of geometry in a particular wire file is built to plus or minus a given value (tolerance). Pro/ENGINEER uses a system of relative tolerance, referring to the fact that the acceptable gap between pieces of geometry is based on the relative size of the geometry.

The default accuracy in Pro/ENGINEER is set to .0012 and the range available is .01 to .0001. Using the default accuracy, the maximum allowable distance between two surfaces when the longest edge of the surface is five inches would be less than $5 * .0012 = .006$ inches. You must create surfaces in StudioTools that adhere to this accuracy to be successful in creating a Pro/ENGINEER solid model.

Whether you are creating a model, verifying a model, or debugging a translation, there are a number of tools within StudioTools to check the quality of the geometry you have created. The most useful tool is the min/max measurement tool in the Locators menu. Use this tool to check the maximum distance between any two surface boundaries.

Get the topology right

The StudioTools stitching operation is recommended to be done on geometry being prepared for transfer to Pro/ENGINEER.

The **Surface Edit > Stitch > Shell Stitch** feature within StudioTools creates a valid solid model topology within the StudioTools modeling environment. The stitching of surfaces within StudioTools greatly improves the robustness of the interface to Pro/ENGINEER. The stitching process also identifies surface boundaries that exceed the prescribed tolerances. These problems can then be corrected by the designer prior to the translation of the data to Pro/ENGINEER.

In addition, the stitching process also identifies duplicate surfaces in the model and orients the surface normals of the completed shell.

When models are constructed using the advance surface tools (swept, Rail Surface and square) it is quite common to create a number of smaller surfaces along the edge of one larger surface. This modeling technique does not create the twin edges required for a solid model. The stitching feature will automatically create the twin edge topology required by Pro/ENGINEER.

One case that cannot be solved topologically is the closed or periodic surface (a primitive sphere is an example of a closed surface). The reason for this is that in most solid modelers, a

face cannot be joined to itself. The presence of closed or periodic geometry in StudioTools (not true for Granite) is another reason that geometry intended for transfer to Pro/ENGINEER must be stitched before export. Using stitch has the same effect as detaching the geometry to create two surfaces before writing the IGES or STEP file for Pro/ENGINEER.

To import StudioTools models into Pro/ENGINEER

Define Absolute Tolerance Process (STEP and IGES only)

Before importing any foreign geometry (such as models created in StudioTools) the Pro/ENGINEER user can change from the default **Relative Tolerancing process** to the **Absolute Tolerance process**. Before a foreign model (that is created anywhere other than Pro/ENGINEER) is imported into Pro/ENGINEER, the desired absolute tolerance can be set to the value that the incoming model was built to.

For example if the Curve Fit Distance in StudioTools was set to 0.002mm, then the Absolute Tolerance in the Pro/ENGINEER work session should be set to 0.002mm.

This option can be enabled by writing the line:`enable_absolute_accuracy yes` into the `config.pro` file of the working directory.

Once the option is enabled you must go to the Setup section of the Pro/ENGINEER application and choose Absolute Accuracy, and set the units and numerical value of the tolerance you wish to work at, each time a new part is created.

This is important to ensure that the StudioTools-created model can be used in further downstream operations in Pro/ENGINEER.

Advanced data sharing techniques

The following are some suggestions for StudioTools modeling that provide enhanced inter-operability with Pro/ENGINEER.

Export individual “Features” from StudioTools

Because Pro/ENGINEER creates each element of a model as a feature, it can be very useful to import components of the StudioTools model as individual export files that can be manipulated in Pro/ENGINEER as individual import Features. Major components of your StudioTools model can be transferred separately so that they can be used to construct individual features within Pro/ENGINEER.

The advantage of this technique is that individual features can be “reordered” in Pro/ENGINEER to give added flexibility to the engineer. The Feature > Reorder command allows the user of Pro/ENGINEER to modify the sequence feature construction. This is useful during the engineering process. Additional “mechanical” features are added to the industrial design model and the result is based on geometry previously created.

Surface replacement

The surface replacement technique can be very useful when the model is a mix of mechanical elements defined by an engineer and styling elements defined by an industrial designer. By replacing the styling elements of a Pro/ENGINEER model all of the parametric/feature information is retained for the mechanical elements. This allows for continued parametric editing, automatic dimensioning, and so on.

Exporting assemblies from Pro/ENGINEER in IGES format

When exporting assemblies from Pro/ENGINEER, there are four types of IGES output available: Flat, One Level, All Levels and All Parts.

Flat

With the Flat option, Pro/ENGINEER exports the entire geometry of the assembly in a single IGES file. All components are transformed into model space before exporting, but there will be no hierarchy contained in the IGES file. All geometry of the assembly exists in the Flat IGES file, and it will all be correctly positioned in model space.

To help organize these files that have the potential of being very large, place each instance of a part within an assembly into its own layer in Pro/ENGINEER before exporting the assembly as IGES Flat. The layers will be transferred as "levels" along with the assembly in the IGES file. The IGES levels are translated into ALIAS Sets. This means that the members of one set are all of the surfaces that comprise an instance of a part in the assembly.

The interface ID that was specified for each layer is the means by which Pro/ENGINEER layer information is transferred via IGES. IGES does not support names for layers. Layers in IGES are called "levels" and a level is identified by a number, not a name. This is why Pro/ENGINEER asks you to assign a number as well as a name to a created layer. The name is more useful within Pro/ENGINEER, but the number is important for data transfer.

When the IGES file is imported into StudioTools, the IGES level information is created as StudioTools Sets. To display the level information, go the Set Lister (Pick > from lister > SETS). You will notice there exists sets whose names have the format LEVEL#n, where n is the "interface id" that was specified in Pro/ENGINEER.

One level

Outputs an assembly IGES file with external references pointing to the IGES files of its components. This contains only top-level geometry.

All levels

With the All Levels option, Pro/ENGINEER outputs $n + 1$ IGES files, assuming that there are n parts or sub-assemblies in the assembly. There will be one IGES file for each part or sub-assembly (for a total of n IGES files) and one master IGES files that contains external references (IGES entity 416 form 1) to the n -component IGES files. Each external reference to a component IGES file within the master IGES file is contained within an IGES Subfigure Definition (entity 308), which is instanced once by an IGES Subfigure Instance (entity 408). The model of each component referenced by the master IGES file is in definition space; that is, placed at the origin. Each component is transformed into model space via the

transformation contained in the Subfigure Instance (entity 408) in the master IGES file.

If the n-component IGES files are individually imported into StudioTools, the resulting model will be incorrect, since each component will be placed at the origin, rather than the correct spot in model space.

If the master IGES file is imported into StudioTools, there will be no model at all! This is because StudioTools does not support the IGES External Reference (416 form 1) entity. This entity is generally frowned upon because it contains the filename of the component IGES file, and filenames are generally not portable between operating systems (for example, Unix > DOS).

All parts

Outputs an assembly to IGES as multiple files containing geometry information of its components and assembly features. These parts use the same reference coordinate system to ease reassembly in the receiving system.

Detailed file format information

PTC Granite format (Windows Only) (page 116)

STEP format (page 121)

IGES format (page 106)

CATIA V4

StudioTools CATV4 DirectConnect is a stand-alone utility that allows the exchange of 3D model data between StudioTools and CATIA using the CATIA/StudioTools neutral format CAI.

CATIA Requirements

- Version 4.2n of CATIA
- SurfaceStudio, AutoStudio, Studio or DesignStudio.

Use the following summarized list of modeling practices discussed in this section as a quick reference guide if problems arise.

Before you create the model

- Units should be set to mm.

In the **Construction Settings** window:

- The **Rational** geometry flags should be toggled **OFF**.

Tolerances should be set as follows:

- **Curve Fit Distance** = .01 mm (lower as necessary)
- **Curve Fit Checkpoints** = 10
- **Max Gap Distance** = .01 mm (this value should remain the same as **Curve Fit Distance**)
- **Trim Curve Fit** = .005 mm (lower as necessary)

While you create the model

- Use degree 5 curves and surfaces to achieve curvature continuity between surfaces.
- Models should be transferred periodically from StudioTools to CATIA during construction to manage the quality of the model being created.
- Periodically stitch the geometry once it is in CATIA to ensure that the model meets all tolerance requirements.
- Avoid using **Object edit > Attach > Attach** since this function creates multiknots in StudioTools geometry.

- Avoid using **Surfaces > Skin** between trimmed surface boundaries, since excessive amounts of data are created in the resulting surface. If Skin is used between trim boundaries, the resulting surfaces should be checked for multiknots before export.
- Use surface building tools such as **Surfaces > Boundary Surfaces > Square** and **Surfaces > Swept surfaces > Rail surface** to ensure and control curvature continuity between surfaces.

Workflow

Before transferring geometry between StudioTools and CATIA, you should consider the purpose of the transfer to plan an appropriate workflow. Two common workflows are:

- Geometry (describing mechanical components) is transferred from CATIA to StudioTools as reference data for concept design surfacing, then the StudioTools model is transferred back to CATIA.
- A StudioTools model is transferred to CATIA, and both StudioTools and CATIA databases are developed independently. Later, the modified StudioTools model is transferred again to CATIA, replacing the StudioTools geometry from the first transfer. In this scenario, all work done in CATIA on the original StudioTools model (ribs, thickness) will be applied to the new modified StudioTools model.

There are many variations on these two examples. Whatever the transfer scenario, you should carefully plan the transfer process, to ensure that the appropriate data is written out and is useful.

What happens when you replace StudioTools geometry

A common workflow using StudioTools and CATIA together is one where you replace existing StudioTools geometry in a CATIA model file with updated StudioTools geometry. This workflow allows you to continue working in StudioTools, modifying a model that has already been passed over to CATIA.

When you want to update the CATIA database with the completed changes, the surfaces that have been modified are

passed to CATIA. You import the new StudioTools geometry and then redefine the skin that includes the faces in question using the **Limit2 > Skin > Create/Modify** tool.

If you want to make changes to a face or surface using StudioTools and then include that modified surface in the CATIA model, you only have to redefine the *skin* to its members. That is, this time you leave out the original face and include the new StudioTools-modified face. This way StudioTools geometry can be used to modify CATIA models at any point throughout the development cycle.

What are the curve to fit distance tolerances in StudioTools

The **Curve Fit Distance** is the tolerance to which trim boundaries are rebuilt to (or approximated). The default positional tolerance in CATIA is .1 mm, and the StudioTools **Curve Fit Distance** setting should be set to 0.01mm.

This **Curve Fit Distance** setting should normally be accurate. If you find that it is not resulting in StudioTools geometry that can be successfully used in CATIA, then experiment with the **Curve Fit Distance**—it can be set to as low as 0.005 mm. This setting will enhance the success of post transfer processes, such as skinning, that are to be carried out once the geometry is in CATIA.



The **Curve Fit Distance** tolerance in StudioTools should not be set at less than 0.001 mm. Lower than this will impact processing time.

Whether you are creating a model, verifying a model, or debugging a translation, there are a number of tools in StudioTools you can use to check the quality of the geometry you have created. The most useful tool is the **Locators > Deviation** Min/max measurement tools. Use this tool to check the maximum distance between any two surface boundaries.

Detailed file format information

CAI format for CATIA V4 files (page 125)

CATIA V5

StudioTools CATIA V5 DirectConnect is a stand-alone utility that allows the exchange of 3D model data between StudioTools and CATIA V5 using the native CATIA part (.CATPart) and product (.CATProduct) documents. Please follow installation instructions provided in the README.txt file.

CATIA Requirements

- Version 5 of CATIA (Release 10, 11, 12, 13, 14, or 15) and an appropriate CATIA license.
- Version 13 of SurfaceStudio, AutoStudio, Studio or DesignStudio, and a CATIA V5 DirectConnect license.
- IRIX or Windows operating system.
- On IRIX, `ftn_eoe` *must* be installed for CATIA V5 DirectConnect to work:
 - ◆ `ftn_eoe` Standard Execution Environment (Fortran Headers and Libraries, 7.4)
 - ◆ `ftn_eoe.sw.lib` Standard Execution Libraries (N32bit)

See *Before you create the model* (page 114) for a list of modeling practices to use as a quick reference guide if problems arise.

If you have a network installation of CATIA V5, please see the installation notes (README.TXT) for important information about environment variables that must be set.

Before you create the model

- Units should be set to mm.

Optimal tolerances should be set as follows, as recommended by Dassault Systemes:

- **Curve Fit Distance** = 0.001 mm (lower as necessary)
- **Curve Fit Checkpoints** = 10
- **Max Gap Distance** = 0.01 mm
- **Trim Curve Fit** = 0.005 mm (lower as necessary)
- **Topology Distance** = 0.02 mm

While you create the model

- Models should be transferred periodically from StudioTools to CATIA during construction to manage the quality of the model being created.
- The StudioTools model should be capable of being successfully stitched before export. If you periodically stitch the geometry to ensure that the model meets all tolerance requirements, you'll have a good indication of whether the final model will stitch correctly.
- Avoid using **Object edit > Attach > Attach** since this function creates multiknots in StudioTools geometry that may result in unusable geometry in CATIA.
- Use surface building tools such as Square and Rail Surface, taking advantage of the **Boundary Rebuild** option to control curvature continuity between surfaces and ensure surfaces do not contain multi-knots.
- Use **Evaluate > Check model** to be alerted to potential problems: it's another good practice.

Workflow

Before transferring geometry between StudioTools and CATIA, you should consider the purpose of the transfer to plan an appropriate workflow. Two common workflows are:

- Geometry (describing mechanical components) is transferred from CATIA to StudioTools for concept design surfacing, then those Studio surfaces are transferred back to CATIA.
- A StudioTools model is transferred to CATIA, and both StudioTools and CATIA databases are developed independently. Later, the modified StudioTools model is transferred again to CATIA, replacing the StudioTools geometry from the first transfer. In this scenario, all work done in CATIA on the first StudioTools model transfer will affect the new, modified geometry.

There are many variations on these two examples. Whatever the transfer scenario, you should carefully plan the transfer process, to ensure that the appropriate data is written out and is useful.

What are the curve fit distance tolerances in StudioTools

The **Curve Fit Distance** is the tolerance to which trim boundaries are rebuilt to (or approximated). The default positional tolerance in CATIA V5 is 0.001 mm, and the StudioTools **Curve Fit Distance** setting should be set to 0.001mm.

This **Curve Fit Distance** setting should normally be accurate. If you find that it is not resulting in StudioTools geometry that can be successfully used in CATIA, then experiment with the **Curve Fit Distance**—it can set to as low as 0.001 mm. This setting will enhance the success of post transfer processes, such as skinning, that are to be carried out once the geometry is in CATIA.



The **Curve Fit Distance** tolerance in StudioTools should not be set at less than 0.001 mm. Lower than this will impact processing time.

Whether you are creating a model, verifying a model, or debugging a translation, there are a number of tools in StudioTools you can use to check the quality of the geometry you have created. The most useful tool is the **Locators > Deviation Min/max** measurement tools. Use this tool to check the maximum distance between any two surface boundaries.

I-deas NX series

You can intergrate StudioTools models into I-deas NX (UGS PLM solutions). Often the workflow will require that the designer send geometry over to the engineer using a CAD system. The CAD operator will then use the model to describe a solid part and perform other engineering process such as describing ribs, bosses and other mechanical details.

At any point the designer may need to update the information in the CAD system by re-exporting changes made to the original model in StudioTools so that those changes can be integrated into the CAD database.

MasterSeries Requirements

- I-deas NX 9, 10, or 11 series.
- I-deas DirectConnect
- SurfaceStudio, AutoStudio, Studio or DesignStudio.

Use the following summarized list of modeling practices as a quick reference guide to avoid data transfer problems.

Before you create the model

- **Units** should be set to mm.
- In the Construction Options window:
 - ◆ The **Modeling Modes** should be set to **NURBS**.
- Tolerances should be set to the following values
 - ◆ **Curve Fit Distance** = 0.005 mm
 - ◆ **Maximum Gap Distance** = 0.005 mm
 - ◆ **Curve Fit Checkpoints** = 10
 - ◆ **Trim Curve Fit** = 0.001 mm

While you create the model

- Use degree 5 curves and surfaces to achieve curvature continuity between surfaces.
I-deas does not support degree 7 geometry. Geometry created in StudioTools which is degree 7 will be rebuilt to degree 3 (cubic) upon import to I-DEAS. Adjacent

StudioTools surfaces which had been built with continuity between them may no longer have that continuity after they are rebuilt in I-DEAS.

- Surfaces can be overbuilt and trimmed back before exporting. This will result in a greater success rate when the surface geometry is offset in I-DEAS.
- Models should be transferred periodically from StudioTools to Master Series during construction to manage the quality of the model being created.
- Avoid using **Surfaces > Skin** and **Object Edit > Patch** between trimmed surface boundaries since excessive amounts of data are created in the resulting surface.
- Use surface building tools such as Square and Rail Surface, taking advantage of the **Boundary Rebuild** option to control curvature continuity between surfaces and ensure surfaces do not contain multi-knots.

Geometry types exported to I-deas

The following geometry types can be exported to I-deas using I-deas DirectConnect:

- Single CVs (points)
- All curves (with or without attributes: lines, polylines, etc.)
- Faces
- Curves on Surface
- Surfaces
- Target Surfaces
- Trimmed Surfaces
- Trimmed Surfaces with multiple trim regions
- Target Trimmed Surfaces
- Shells



Polysets are the only geometry entities that are not supported by Master Series.

Workflow

The workflow of transferring data into I-DEAS NX series requires that the designer send geometry over to the engineer. The CAD operator will then use the model to describe a solid part and perform other engineering process such as describing ribs, bosses and other mechanical details.

At any point the designer may need to update the information in the CAD system by re-exporting changes made to the original model in StudioTools so that those changes can be integrated into the CAD database.

Shell imported geometry

Shelling, or creating a topological description from StudioTools models, is a common workflow in StudioTools and I-deas NX series. When exporting a model from StudioTools, you must ensure that the geometry is built to the correct tolerance and that it can be stitched. The stitching process in StudioTools identifies gaps between surfaces so that you can repair the appropriate geometry before exporting to I-deas NX series.

During stitching, the surfaces are twinned. This means that the surface boundaries may be split to accommodate adjacent surfaces, and periodic geometry is detached into multiple surfaces.

For this reason, you should save the StudioTools wire file before stitching so that if further modifications are required to be made to the StudioTools model, the construction history will be intact.



Stitched geometry saved to an StudioTools wire file cannot be unstitched to its original state.

Export features from StudioTools

The term *part* in I-deas NX series refers to geometry that has been saved out as a part. The term *feature* refers to any attribute that augments the basic shape of a part and distinguishes it from other parts that could be derived from the same basic shape. More precisely, features are objects whose key dimensions and orientations have been controlled,

thus allowing you to control your design. You can bring in parts or features to replace existing features on the workbench.

Since I-deas NX series creates each element of a model as a part or feature, it is useful to import components of the StudioTools model as individual StudioTools wire files that can be manipulated in I-deas NX series as individual features. Major components of the StudioTools model can be transferred separately so that they can be used to construct individual features. The advantage of this technique is that individual features can be “replaced” to give added flexibility.

Unigraphics

StudioTools Unigraphics DirectConnect is a stand-alone utility that allows the exchange of 3D model data between StudioTools and Unigraphics.

- **See movie:** unigraphics.rm

Unigraphics Requirements

- In order to use Alias UG DirectConnect, the Unigraphics installation must include one of the following licenced options:
 - ◆ Unigraphics NX Open API Execute or
 - ◆ Unigraphics NX Open Package Execute
- You must have access to Unigraphics software for the same platform on which you are running StudioTools.
- Unigraphics utilities supported now include versions 16-20, NX, and NX2 for AIToUG and UGToAI.
- SurfaceStudio, AutoStudio, Studio or DesignStudio.

Before you create the model

- Units should be set to mm.

In the **Construction Settings** window:

- ◆ The **Rational** geometry flag can be toggled **OFF**.

Tolerances should be set as follows:

- ◆ **Curve Fit Distance** = .01 mm (lower as necessary)
- ◆ **Curve Fit Checkpoints** = 10
- ◆ **Max Gap Distance** = .01 mm (this value should remain the same as **Curve Fit Distance**)
- ◆ **Trim Curve Fit** = .005 mm (lower as necessary)

While you create the model

- Use degree 5 curves and surfaces to achieve curvature continuity between surfaces and successful data transfer.
- Models should be transferred periodically from StudioTools to Unigraphics during construction to manage the quality of the model being created.

- The StudioTools model should be successfully stitched before export, but you should also periodically stitch the geometry to ensure that the model meets all tolerance requirements.
- Avoid using **Object edit > Attach > Attach** since this tool creates multiknots in StudioTools geometry.
- Avoid using **Surfaces > Skin** between trimmed surface boundaries, since excessive amounts of data are created in the resulting surface. If **Skin** is used between trim boundaries, the resulting surfaces should be checked for multiknots before export.
- Use surface building tools such as **Surfaces > Boundary Surfaces > Square** and **Surfaces > Swept surfaces > Rail surface** to ensure and control curvature continuity between surfaces.

Workflow

Before transferring geometry between StudioTools and Unigraphics, you should consider the purpose of the transfer to plan an appropriate workflow.

Two common workflows are:

- Geometry (describing mechanical components) is transferred from Unigraphics to StudioTools to be used as reference data for concept design surfacing, then the StudioTools surface model is transferred back to Unigraphics.
- A StudioTools model is transferred to Unigraphics, and both StudioTools and Unigraphics Databases are developed independently. Later, the modified StudioTools model is transferred again to Unigraphics, replacing the StudioTools geometry from the first transfer. In this scenario, all work done in Unigraphics on the first StudioTools model transfer will affect the new, modified geometry.

There are many variations on these two examples. Whatever the transfer scenario, you should carefully plan the transfer process, to ensure that the appropriate data is written out and is useful.

Detailed file format information

Unigraphics proprietary format (page 111)

Solid Imaging

Solid Imaging is a component of Rapid Prototyping which uses a database to translate three-dimensional geometry into physical models or parts using a variety of resins and other materials. The file formats used by StudioTools to output files for Rapid Prototyping are the STL and SLC.

Solid Imaging Requirements

NURBS surfaces must be translated into either the .stl or .slc format before reading the file into the solid imaging machine software.

Workflow

StudioTools wire files exist as NURBS data. To use that data to create physical models using solid imaging technologies, you must translate the NURBS to either the .stl or .slc format so that the geometry can be read by the solid imaging machine's software. Included in the list of solid imaging technologies is SLA (Stereolithography), SLS (Selective Laser Sintering), LOM (Laminated Object Manufacturing), SGC (Solid Ground Curing), FDM (Fused Deposition Modeling) and others.

Converting the StudioTools geometry to the .stl format or the .slc format can be done from within StudioTools.



Consult with the operator of the solid imaging machine to optimize the transfer of data.

STL Format

An.stl file is a tessellated file (binary or ASCII), which means the NURBS surface is described by a series of triangles. The resolution of this polygonized data base is defined in StudioTools by the subdivision characteristics of the original NURBS surface. Once the tessellated geometry is sent to the Solid Imaging technology, the geometry is sliced, and then those slices are used to describe the physical model that will be produced.

With STL as the transfer format, you can send geometry to most Solid Imaging technologies while controlling the resolution of the finished model.

The STL file exported from StudioTools conforms to 3D systems file format version 2.0. When you export a model as an STL file, StudioTools displays:

- A "solid check" is run on the model and the results are displayed at the prompt line. This tessellation check determines if it is a valid solid watertight model or if it has any gaps indicating topological errors. This allows the StudioTools user to determine if the data being transferred can be used by the operator of the solid imaging (for example SLA) machine to build the part.

If gaps are found, the user receives a warning indicating that it is an illegal solid and the number of free edges in the model. When you view the model, edges with gaps are highlighted in red so that you can easily identify where gaps are and then repair the surface model.

- Stitch and tessellation tolerances options, allow you to set the merge vertices tolerance, the maximum distance at which two vertices will be merged together into one.
- During tessellation, degenerate triangles (with two or more equal vertices) are removed and the normals of the triangles are recalculated.

SLC format

An.slc file (StereoLithography Contour) cuts 2D contours of the 3D data base. These contour lines are polylines. The advantage to using this file format is that the NURBs geometry description in StudioTools is directly sliced and therefore fewer iterations are required between the original geometry and the data sent to the Solid Imaging machine to be built.

SLC header information

The header section of the .slc file is an ASCII character string (up to 2048 bytes) containing global information about the model.

The output in the header provides the following information:

- SLC file format version number (-SLCVER2.0)
- Output units (-UNITS<INCH/MM>)
- Type of model (-TYPE<PART/SUPPORT/WEB>)
- Vendor package and version number (which produce the SLC file (-PACKAGE ALIAS STUDIO V12.0)
- Calculations and sets from SLC output x,y,z extends of the model (-EXTENTS m_x,M_x,m_y,M_y,m_z,M_z)

Header keywords (CHORDDEV, ARCRES, SURFTOL, GAPTOL, MAXGAPFOUN, EXTLWC, STHICK, STARD and ENDD) are set to 0.0.

TC VisProducts

TC Direct Connect is a real-time rendering solution based on OpenGL. It is a file-based translator used to bidirectionally convert StudioTools native format wire files to UG's native formatted DirectModel files in the Jupiter (.jt) file format.

Functionality of TC VisProducts

The following sections provide detailed descriptions of the functionality available in StudioTools's TC Direct Connect translator. At a high level, there are two interfaces available in TC Direct Connect. The software is offered either as a stand-alone command-line driven tool, or it can be launched from within StudioTools via the **File > Save As** □ > **TC VisProducts** menu. The TC software is designed to operate in one direction, converting StudioTools data to TC DirectModel files. (For import into StudioTools, please see the JT import translator.)

How the TC translator works

Hierarchy in StudioTools is represented by DAG node objects. The basic parent/child relationship is reflected through the group node, which refers to a list of child DAG nodes. This type of DAG node allows the hierarchical grouping of DAG nodes. A group node can share its list of children with another sibling group node.

The translator works by converting each DAG node object into the equivalent DirectModel node object. DirectModel can represent many of the same types of nodes that StudioTools supports. For example, StudioTools surfaces are individually tessellated into unique DirectModel parts. There should be a 1:1 correspondence between StudioTools surface names and DirectModel part names. If surfaces are grouped in StudioTools, they will also be grouped in DirectModel.

The TC translator is controlled through several possible methods. You have the choice of either an StudioTools plug-in, accessed within the **File > Save As** □ > **TC VisProducts** menu, or batch mode driven by a command-line. This gives you control over various tessellation parameters and other important options.

Index

Numerics

2D contours
cutting in 3D data (SLC) 125

A

accuracy
defining for solid modelers 99

action window 81

actions
and channels 76
base actions & time warps 76
copying & pasting 81
creating 73
definition 77
motion path 76
parameter curve 76
sharing between channels 81
types of, in animation 76

animatable items 69

animating
basics of 68
DAG nodes 69
definition 68
objects 73
parameters 75
using expression channels 78
verifying 75

animation parameters
and channels 77
definition 77
expressions 78

autofly tool
camera eye 79
camera up vector 79
camera view 79

B

base actions
in animation channels 76

basic animation 68

C

CAD data transfer
rational and non-rational geometry 99

cameras
camera eye 79
camera up vector 79
camera view 79

CATIA formats
exchanging 3D data with 111, 114
summary of modeling practices 111, 114
using skins 113

channels
actions 73
and animation parameters 77
creating 73
definition 77
description 72
expression channels 78
many or one action 76
one-to-many relationship 76
using many actions 76
verifying 75
viewing particular actions 77

copying
actions 81

creating
a solid model of a cube 96
channels for animation information 73
model guidelines 111, 114

D

DAG node
definition 127

DAG nodes
animating attributes 69

E

exporting
solid model of a cube 96
surfaces(faces) 96
tessellated files 124

expressions
entering 78

F

feature
term in I-deas NX series 119

fillets
creating rational 99

Fused Deposition Modeling (FDM) 124

G

geometry
checking surface continuity 100
getting it right for data transfer 99

I

I-deas NX series 117–118
modeling practices 117

L

Laminated Object Manufacturing (LOM) 124

linear units
base for data transfer 99

M

meshes 63

how they differ from
polysets 64
operations allowed on 64

modeling

before you create a model
(CATIA) 111, 114
for CATIA 111, 114
for I-deas NX series 117

motion path

actions 76
of camera 79

N

NURBS

and tessellation 124

O

objects

animating 68, 73

one-to-many relationships 76

P

parameter control window 75

parameter curve actions 76

definition 73

part files

term in I-deas NX series 119

pasting

actions 81

R

rational and non-rational

geometry

for CAD 99

S

Selective Laser Sintering (SLS) 124

shelling

in I-deas NX series 119

skins

using in CATIA 113

SLC formats

2D contours 125
format options 125–126

Solid Ground Curing (SGC) 124

Stereolithography (SLA) 124

stitching

in I-deas NX series 119

surfaces

NURBS tessellation 124

T

TC DirectConnect 127

definition 127
how the translator works 127

tessellated files

exporting 124

time warps

in animation channels 76

tolerances

standards for CAD
systems 99

translators

TC DirectConnect 127