Seal of Good Modeling 2

A.1 Avoid Complex Hierarchies 2

A.1.1 Replace Glue as Child links with Glue as Sibling links in older models 2

A.1.2 Use Object Union for groups of objects that do not touch 2

A.1.3 Don’t Group All Objects Into One 3

A.2 Triangulate 5

A.3 Reduce Your Poly Count 6

A.4 Activate and Deactivate the Bridge 7

A.4.1 Bridge basics: 7

A.4.2 Player and Model libraries 7

A.4.3 Workflows: 7

A.4.4 Bridge limitations: 8

Seal of Good Modeling

While we at Caligari try to make the trueSpace as robust as possible, it never hurts to develop good working practices for yourself, too. You will get the best out of trueSpace if you work with the program’s strengths and avoid unnecessary complexity. Just the smallest amount of thought, planning, and workflow adjustments can reap major benefits!

This section aims to give you some pointers and hints on how to develop scenes and models to maximize what trueSpace can do for you. Many of these tips will help with scenes that were originally created with an earlier version of trueSpace where such issues still impacted performance, but that effect can be more noticeable when using the real-time renderer or the bridge. With some thought and planning, the real-time Player can help you work fast and smooth with models that have so many polygons they could bring other displays to a halt!

A.1 Avoid Complex Hierarchies

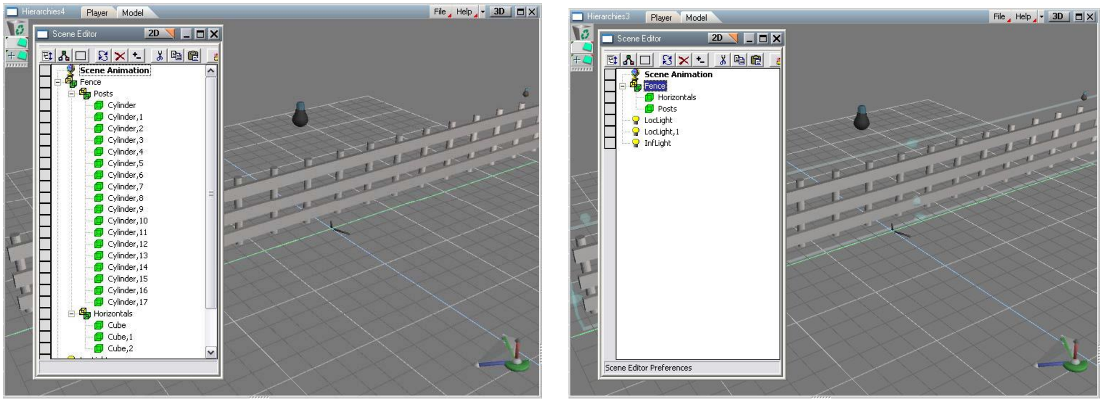
A.1.1 Replace Glue as Child links with Glue as Sibling links in older models

Glue as Child is functionally equivalent to Glue as Sibling, but it tends to create much deeper hierarchies. Deep hierarchies kill performance in all versions of trueSpace, so we removed Glue as Child from trueSpace altogether. If you have some models created in an older version of trueSpace that have a lot of Glue as Child links, you can improve performance significantly by ungluing them and converting them to Glue as Sibling links.

A.1.2 Use Object Union for groups of objects that do not touch

Note here that the hierarchical structure is the problem, not the number of faces or vertices. You can simplify the hierarchy by using boolean addition of objects that do not touch with the Object Union tool.

In discussions about modeling the advice often given is to avoid using booleans, and this is good advice where geometry touches or intersects as you can get artifacts in the resulting join. However, where the objects do not touch, you end up with no such problem and the boolean is perfectly safe to use.

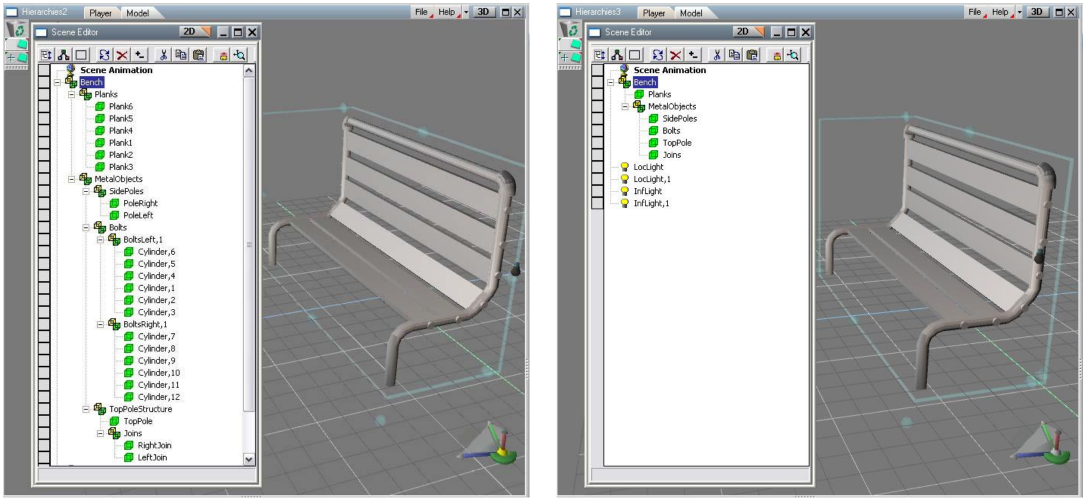


Left shows wide hierarchy using Encapsulate / Glue; Right shows simpler hierarchy using Object Union

A prime example here would be in constructing railings, as seen in the images above. Each post in the railing is a single object, and this could lead to a great number of objects in your scene. If you glue them all together, then the result is a very wide hierarchy, and this could be problematic. Using Object Union to join them together however results in one single object containing all the posts, with no hierarchy at all. This is much easier for trueSpace to handle.

This step is one you might take toward the end of the project, once all the posts are in place and you don't expect to edit the individual objects inside the group again. Even so, the step is not irreversible, as it is possible to separate all the posts again using the Decompose Into Objects tool. Note that this tool does not work if objects that touch or intersect have been booleaned together, but it will work just fine where the objects did not touch or intersect.

This same technique is useful if you plan on exporting your model to other applications, since many other applications do not support the hierarchies that are possible in trueSpace. This makes reducing hierarchies a good modeling practice in general, not just something to be done due to the bridge or real-time renderer.

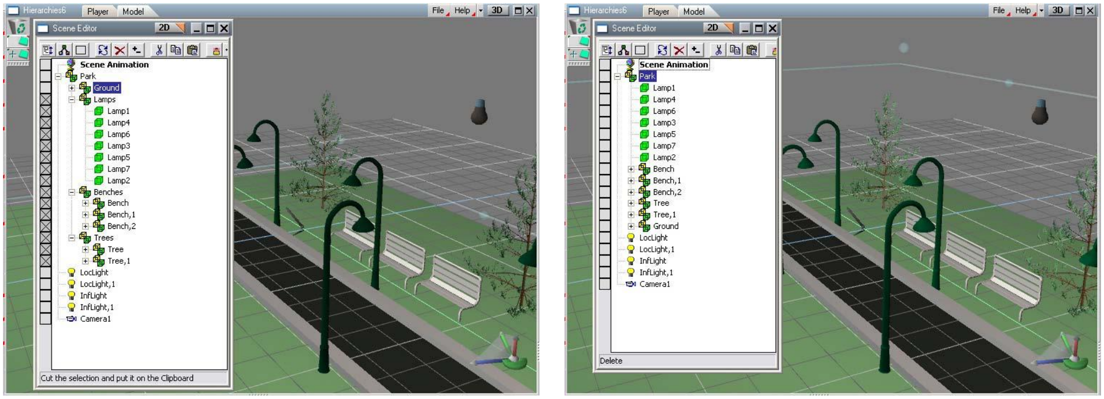


Applying the Object Union approach on a more complex hierarchy gives even greater simplification

A.1.3 Don't Group All Objects Into One

It can be tempting to group everything together into one "super object" - for example if you create a park scene, you may have ground, hills, bushes, railings, people, trees and benches. You could create a structure that had a Park group containing all of those, with the Benches group containing each individual Bench, where each Bench was a group of all the separate parts arranged into groups such as Legs and Back, with groups inside those for Bolts and Planks...

This could be a very complex hierarchy indeed! It may be best to keep the groups separate to avoid a hierarchy that is too wide or too deep, for example each bench could be simplified as shown previously, but then each bench could also be kept as a separate object, rather than glued into a group of all benches in the scene. Should you want to move all objects at once, you can do so via selecting them with a bounding box (click and drag) or by holding CTRL and clicking to select each object. Alternatively, if you frequently move all the objects at once, you could simply glue all the park objects into one park without sub-grouping them into "Trees," "Benches," "People" etc.



Avoiding sub-groups can make a big difference to performance

You can see above that the hierarchy is marginally less attractive in some ways once simplified in the right hand image, since it now has all benches, all lamps, all trees etc shown individually inside the Park object. Yet despite this slightly more ungainly look, this should not remove any degree of usability from the hierarchy, while applying this approach wherever possible will give you a performance in trueSpace.

In combination with the use of Boolean Addition for non-touching objects, this can lead to a greatly simplified hierarchy, making an object or scene much more “digestible” for the real-time renderer and bridge. Note that for compatibility with previous versions of trueSpace, it is best to edit the hierarchies before saving the scene from the previous version and loading it into the current version.

Of course, all example scenes shown here are so simple that they will perform equally well with the “bad” and the “good” approaches to hierarchies. They have to be kept simple, however, to illustrate the concepts. Not only that, but taking care of these details at this sort of simple level will add up as a scene grows in size with more and more objects!

A.2 Triangulate

The real-time renderer requires models to be triangulated, and this happens to all objects in a scene when they are passed across the bridge to the real-time render engine. It often helps to run the triangulation yourself on the source objects in the trueSpace Model window, as then the software does not need to triangulate it for you each time the object is sent over the bridge. Since triangulation can make working with the model harder, this step is best taken at the end of a project, for example if you are about to make a scene available through truePlay, or use the real-time renderer to set-up lighting once modeling is complete.

A.3 Reduce Your Poly Count

The Polygon Reduction tool is greatly improved compared to previous generations of trueSpace. With the quality that can be achieved using the Polygon Reduction tool, you can lighten the load during rendering or for a real-time use of an object or scene without a loss of quality. Choose the level of reduction that matches just how much detail needs to be seen in the object - for instance, if you are using a car model in an animation and it just driving by as part of the background, then there is no need for it to retain high detail.

The tool actually keeps the high poly version stored inside it, and allows you to change how much detail is shown in the real-time renderer through the use of a slider, making it non-destructive. This is ideal for real-time use of objects, and lets you tweak the amount of simplification you use until you are happy with the result.

Of course, with the high poly count version preserved “inside” the object, memory requirements will remain higher. If you want to reduce those requirements too, rather than just lighten the load for the real-time display, then you can permanently extract the low poly version and use that instead, saving the high poly version to a library for safe keeping.

A.4 Activate and Deactivate the Bridge

The Bridge brings you the advantage of being able to use tools, workflows and plug-ins from previous generations of truespace, while also being able to use all the new features in trueSpace. To give you this advantage, though, the Bridge has to do quite a bit of processing, and use quite a bit of memory, and there are times when you simply don't need the Bridge - and that's why you are not forced into having it active all the time.

If you take the time to learn a little about the Bridge - all you need to know is in the next few paragraphs - then you'll be able to use it to your best advantage, ensuring it always helps you.

A.4.1 Bridge basics:

The purpose of the bridge is to synchronize Player scenes with Modeler scenes and synchronize some settings. It is good to remember the following:

- If you make an action in Player, then it takes time to appear in Model, and vice versa.
- It takes more memory to keep both Views synchronized.
- Not all the objects are transformed to 100% equivalent between both views.
- The bridge stays active in Auto mode if at least one Player view (LE, View3D, Object library, Scene library) is opened, and at least one Model View (KFE, main window) is opened.
- The bridge is activated for one-time synchronization each time you are switching between Model and Player View, or if you render with selected VRay renderer.
- If the bridge is activated, it has to synchronize both Model and Player Views. To do this, it has to know the sync. direction (Model scene syncs to state of Player, or vice versa). Because some objects are known only to Model or only to Player, you can lose some objects during synchronization. This can happen mainly if you edit object **hierarchy, mesh, or materials**.

A.4.2 Player and Model libraries

- Player and Model have separate libraries. Be careful to understand the differences mainly for objects and scenes.
- Player libraries can store any object you want, independent of the hierarchy in which you are saving it. This is not possible in Model, where only the TOP hierarchy level of objects can be stored
- If you save a Player scene, then it saves automatically all info from the Model scene too, so you will not lose any information.
- If you save a Model scene, it saves info from Player scene too, except unconnected connectors on the scene object. This means all the info about scene content is saved except items such as Player camera entry point, etc.

A.4.3 Workflows:

- **KF Animations:** If you want to create a key-frame animation without Player scripting, it is best to turn off bridge and work in Full layout. If you need Player materials, then set the bridge switch to “Auto” or “On,” and turn off “Support Model procedural animation.”
- **KF Animation with scripts:** If you want to combine KF animation and Player scripts, then set the “Support Model procedural animation” to “On,” together with bridge to “On.”
- **Modeling:** If you want to just model some new objects without behavior or KF animation, then use Full

layout or turn the bridge off. You can speed up response on the Model side if you turn off “Support Model procedural animation.”

- **Working With Imported Scenes/Hierarchies:** Imported scenes, either from other applications or from earlier versions of trueSpace, may have material or hierarchy issues that need correcting in order to gain optimal performance in trueSpace7. Since these problems cause slow response from trueSpace, it can be hard to correct them! The solution though is simple. With the Bridge set to Auto, switch to the Full layout. This will allow you to change the hierarchies or materials without the Bridge making any updates (since there is no Player or Link Editor window active), and will give you fast performance. Switching back to the Default layout (or any layout that includes a Link Editor or Player window) will cause the Bridge to move your changes from the Model window over to the Player window also - this means the switch may take a moment as all your changes are moved from the Model window you were working in to the Player window. Once the new Layout is loaded, you can save your scene to the Player libraries, and indeed you should find you can work with the scene in a Model window under the Default view as the performance should be improved due to the changes made to the problem areas of the scene.
- **Import objects, scenes, configurations or projects from older trueSpace versions:** If you are in Full layout, or the bridge is turned off, then trueSpace will behave identically to trueSpace6.6. For objects and scenes, there is no difference. For .CFG and .PRJ files, the interface changes are not loaded, and they are replaced by Player ones.
- **Slow Modeling plus behaviors:** Turn Bridge off, do modeling work e.g. type a lot of 2D text or edit hierarchies that could run slow if the Bridge was on. Enable Bridge, Sync from Modeler, and add behaviors in Player. Work in Player and use all of its tools with the Bridge on, as none are known to run more slowly due to the Bridge. Disable Bridge to return to Modeling. Also, reducing texture resolution for textures transferred from Modeler to Player (in the Display Options panel) can significantly increase speed and reduce memory requirements of material-related operations in Modeler with Bridge turned on.
- **Slow Player:** If for some reason Player becomes slow you can turn Bridge off, do modeling work in Model, enable Bridge, Sync from Modeler, then disable Bridge again. Now add behaviors in Player, enable Bridge, Sync from Player, then Disable Bridge. Return to Modeling.

A.4.4 Bridge limitations:

- All Model procedural objects are transformed to Player as special objects (metaballs, plastiform, IK, some materials, arrays, skinning, NURBS, SDS), which behave as polyhedral objects.
- If you modify a hierarchy, mesh, or materials on the Player side, then these objects are converted back to Model View as simple polyhedrons. So if you modify, for example, an IK object in Player, then you can lose the IK structure. No undo is available.
- Player supports some functionality that Model does not support; therefore those objects are not transformed to Model (for example, all script nodes). If you modify a hierarchy or type of an object that contains a script in Model, for example, then it will be transformed back to Player without the script. In this case, Undo is not available.
- Be careful if you load a large object or scene from tS6.6. If the object uses 200Mb in memory, then it can take 400Mb or more if you turn on the bridge. It is important to decide which layout you use in these cases, as some layouts will not activate the bridge, and some will.
- If the bridge is set to Auto mode, it is not recommended to switch between layouts too often. If you use up all your memory, you may be forced to close trueSpace with your task manager.
- If you select the V-Ray renderer, then be aware that the bridge will be activated for each render, so the memory requirements will be higher.

Due to the fact that Modeler's scene is cleared (to save a memory) when Modeler's views are closed and you are in "Auto" state, these limitations should be noted:

- If you turn off bridge while you are in Auto, and none of Modeler's views are displayed, then remember the Modeler's scene will be empty. So if you switch from Player to Modeler and the scene will be empty, just switch bridge to "Auto" or "On" state and choose in synchronization dialog "Sync from Player". Then you can turn off the bridge again.
- If no Modeler's views are open and you choose "On" and then "Off" or "Auto" state, the Modeler's scene will be again cleared.
- Similarly, be careful when you choose "Auto" state with Modeler's views closed. The Modeler's scene will be cleared.
- To copy and then turn off bridge you must do the following: Choose Dual layout (or open at least one Model and one Player view), and turn off the bridge so both scenes become independent. You can then close Modeler's or Player's views as you want, but be careful to the "Auto" state.
- When you have unsynchronized scenes, remember you have to save Modeler's items to Modeler's libraries and Player's to Player libraries. In each case you lose Player's or Modeler's part of the objects. To save all the parts you have to activate bridge and merge/synchronize the content
- If you close the program and the scenes are unsynchronized, Player's scene will be saved automatically to default.ctx, and for the Modeler's scene you will be asked.
- The synchronization dialog appears only when you switch bridge from "Off" state to "Auto" or "On". If you choose "Auto" then the dialog appears only when the bridge is going to be activated.