Object of the exercise

Benjamin Woolley shows you how to achieve dramatic results when creating complex objects.

In the very first *Hands On 3D Graphics* column, I described the building of a piece of virtual architecture using "Boolean" operations. Booleans are essential to creating many complex objects, but can be complicated to use. So I thought, as it is so long since their first appearance in these pages, they might be worth a more detailed consideration.

Booleans get their name from George Boole, the British mathematician who in 1847 made the connection between mathematics and formal logic that is now such an essential feature of computing. The connection between Boole's mathematical tooles (sorry, *tools*) and 3D graphics is an obscure one, and not really worth exploring. The point is that they provide a method for combining 3D objects — "operands", as they are called in a Boolean context. There are three types of Boolean operations: add (sometimes called "union"), subtract (or "difference") and intersect. When you add two operands together, two objects are joined to form one at the points where their surfaces intersect. Polygons that overlap are deleted. Intersect deletes all the polygons in both operands except those that overlap. Subtract removes the volume of one operand from the other.

Using a combination of these three operations, you can create very complex objects very quickly. The subtract operation, for example, can be used to chip away at an object, as a sculptor might a piece of stone. The intersect operation is like moulding, using one shape as the die, the other as the moulding material. The add operation is like welding, enabling you to build up complex shapes by sticking together simpler ones. There are problems with Booleans, though. You have to treat them with great care, otherwise you might end up breaking objects rather than creating them. The most basic problem with Booleans is that they change the actual geometry of the operands, the objects being operated upon. An add operation between two overlapping objects — for example, two cylinders at right angles to one another to create a sort of cross — results in a single object that is superficially identical to the originals from which it was made.

On the edge

But you will soon find that when you start to inspect the underlying wireframe mesh, a lot has changed. To begin with, the software will have created new edges where the shapes once intersected, and these could be untidy, and in the process removed all texture mapping information (i.e. the information that describes how a 2D texture should be wrapped around the 3D object) will have been lost.

Often, the software will turn your lovely original into something that looks like it's been through a metal crusher

Therefore, if the operands with which you started had complex maps, perhaps intrinsic ones generated when you originally built the objects, you will have to recreate them. Another challenge you will face with Booleans is ensuring that the resulting geometry hangs together. Quite often, the software will turn your lovely original into something that looks like it's been through a metal crusher. The most common cause of this sort of problem is where faces or vertices are almost identically aligned, so the software cannot decide which co-incidental elements should be combined, which preserved, and which deleted. Subtractions that involve punching a hole in an object can be particularly difficult if the meshes are complex. Quite often you will find a stray face or even marooned vertex left behind in what was supposed to be empty space.

Subtractions produce another problem: unexpected results. Unlike an add or intersect operation, the result of a subtraction is dependent on the order in which you do it — which operand is to be subtracted from which. You will find yourself getting the order wrong, and producing the exact opposite of what you wanted. Once you are aware of such problems, though, you can start to deal with them. The most important precaution is to ensure that, before performing any Boolean

BABY LOVE

From the moment I first saw it, I knew "Baby Cha-cha" was destined for greatness. With the original release of "Biped", a plug-in for 3D Studio MAX designed to help animate bipedal characters, came a file which showed how to copy the steps or movements used to animate one model, to another. This was demonstrated by applying the cha-cha step originally developed to animate a model of an adult dancer, to a baby. The result was a sensation, a tiny toddler wearing a nappy swinging its hips with all the abandon of a Travolta impersonator. Now, following Baby Cha-cha's appearance on Channel 4's *Ally McBeal*, Autodesk's PR company has put together a sequel and here are some of the frames from it. This time the baby really gets into the swing of things, doing everything from aerobics to kickboxing.

operation, you keep copies of the original objects, either in a separate file or, if the software supports it, in a clipboard. With many 3D authoring packages, it is usually impossible to recover the original geometry if things go wrong.

☛ **Having a backup** of each object operand can pay dividends when it comes to building new map co-ordinates for the object you have created. For example, some software allows you to copy, or acquire mapping co-ordinates from one object and apply them to another. If this is the case, you could simply copy the co-ordinates from a backup of one of your original objects and re-use those.

☛ **A much bigger challenge** is dealing with Booleans that do not work because the geometry of one of the operands clashes with the geometry of the other. One method of dealing with this is to prepare the operands for the operation about to be performed upon them.

☛ **Make sure** that the mesh of each object is properly constructed, that there are, for example, no stray faces or overlapping edges. These will arise if you have already edited the mesh, so keep any changes you make to any of the operands to an absolute minimum, and always carefully check the geometry after each change. Better still, start with primitives (shapes generated automatically by the software), as they will have valid and simple geometry and are likely to behave in a far more predictable fashion when you perform Booleans upon them.

Snap to it

One good trick is to build up a sort of toolset of basic objects to use in Boolean operations: a cylinder to punch holes, for example, a prism to cut grooves, or lettering made with a diamond-shaped cross-section to engrave words. Use alignment and snap commands to make sure the tool is accurately positioned with respect to the object being operated upon. If, for example, you are using the pyramid to cut a groove of a fixed depth in, say, a block, make sure the pyramid's "cutting" edge is exactly parallel with the surface to be grooved.

One final consideration is to make sure that you do not use Booleans except where they are necessary. Add operations can often be avoided simply by grouping the objects that were going to be added together in some other way. Sometimes it's a good idea to use subtract and intersect operations to create a rough template of the geometry you are after, and then hand-craft a finished version.

Don't be put off experimenting with Booleans — they can produce dramatic results with minimum effort. Just make sure that you handle them with care.

PCW CONTACTS

Benjamin Woolley *can be contacted via the PCW editorial office (address, p10) or email* **3dgraphics@pcw.co.uk**

▶ **Stills, complete with motion-blur, from the Cha-cha baby's new animation. Next stop, Hollywood**