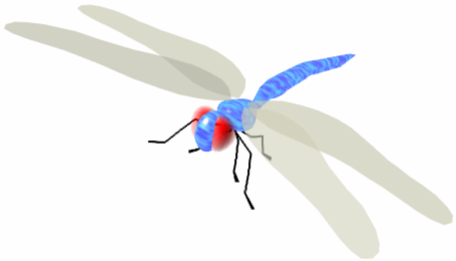
Wave

A Plain English

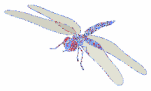
Jac



e-mail: waveguide@sky.com

WaveGuide

An Introduction to LightWave on the Amiga



LightWave is regarded by many people as the Amiga's premier 3D rendering system. The following is a comprehensive guide to its functions and operation. You could call it a 'plain English' rewrite of the rather quirky manual provided with the NewTek package. The information contained here is applicable to LightWave running on different platforms, though it was prepared using version 3.5 on the Amiga. Subsequent versions have expanded capability and some additional features, like *Inverse Kinematics*, *MetaNURBS* and *Plug-ins*, which are not included in this account. You must also understand that there is considerably more to the art of 3D modelling and animation than I can describe here. This guide concentrates on the operation of LightWave's interfaces, known as *Layout* and *Modeler*. You may have already discovered that using Modeler with competence is almost impossible without an easily understood guide. On the other hand, most people will find Layout easier to work with because its functions are pretty well self-evident. Information on both Layout and Modeler is given here in sufficient detail to enable their full potential to be explored and eventually mastered. You will also find 'Crash Course' notes to help you to dive straight into this superb piece of software. They will help you to get quickly under the skin of the programme. They should, for example, enable you to produce animated three-dimensional text, plus lens flares, in no time at all. While NewTek's original manual is comprehensive to a fault, parts of it are poorly compiled and contain blatant errors, though some are less obvious. Also, their phraseology may be an acquired taste, but I found several of their explanations quite confusing. I hope this effort is better, but for a more comprehensive insight on the techniques used by LightWave professionals, I suggested you buy or borrow one of the books that have been published. An example is *LightWave Powerguide* by Dan Ablan (ISBN 1-56205-633-6), though there are many others. LightWave is in continuous development and guide books are published at regular intervals. A useful source of information will be found at *www.flay.com*, a website specialising in all things LightWave. And don't forget to check out the WaveGuide website at *www.intuitionbase.com/waveguide* where there's additional material, tutorials, downloads, etc.

It is presumed that you are at least familiar with the concepts involved in 3D modelling and animation. Indeed many Amigans began their venture into 3D rendering with Imagine (Impulse Inc.). Version 2 was given away on a coverdisk of *Amiga Format* in 1993 and Imagine is evolving still, thanks to its many devotees. There are several other popular systems of course and mention should be made of Real 3D, Cinema 4D, Aladdin 4D and Tornado 3D. Each has its supporters within the Amiga community and in recent times, amongst PC users too. LightWave first appeared in 1987 as part of a combined hardware/software, NTSC workstation known as the 'Video Toaster'. The Amiga was at its heart and essential for its operation. No other computer of the day could provide the necessary video capabilities at such a low cost. However, use of the Toaster was confined more or less to America and within the professional video sector. Not withstanding, the LightWave rendering system began to achieve greater and greater public recognition through the cinema and through tv and films like Robocop, Babylon 5, SeaQuest DSV and Star Trek TNG. These presentations showed how fantastic 3D animations may be produced using LightWave and the Amiga computer. Nowadays, we see them regularly on television in the shape of advertisements. Many tv ads are marvels of 3D animation, which even amateurs are capable of achieving. For example, an animated bottle of Listerine mouthwash appeared on American tv for ages. In the UK, you may be more familiar with the toothpaste man in the Colgate advert. These and many more were created with LightWave. In late 1994, following years of anticipation, NewTek Inc. released the software into the Amiga market. This was no doubt encouraged by the appearance of a semi-official, PAL compliant incarnation called LightRave (Warm and Fuzzy Logic Inc.). LightRave consisted of forty-six (!) floppies and a 'dongle'. The latter was a hardware gizmo which fooled the Amiga into believing it was a Video Toaster. NewTek's version 3.0 enabled either PAL or NTSC Amigas to operate independently of the Toaster hardware. The box contained the software, another dongle, the manual of which I speak and a help video. Versions 3.1 and 3.5 followed shortly after and the rest, as they say, is history.

Installing the Dongle

The dongle supplied with LightWave ensures that the software will only run on one machine at a time. It's a small plastic block with a through-connector for the parallel port and contains an encoded switch which the software must recognise. Use of the programme without the dongle will cause any attempt to open Modeler to quit to Workbench. With the computer power off, remove any cable attached to the parallel port. Insert the dongle into the port and secure it with the mounting screws. Your printer or other parallel device may now be connected to the through port. Other than enabling the LightWave programme, the dongle has no effect on any attached equipment. It does not require an attached device to function. You may now power up the computer.

Installing the v3.5 Software

You should have seven floppy disks named *LightWave Disk 1* through *LightWave Disk 7*. These contain the LightWave 3D code in a compressed state. *LightWave cannot be run from the floppies*. Version 3.5 can be installed to any Amiga computer (except A1000) fitted with a Motorola 68000-series processor. The minimum recommended is the 68030 and the 68060 is ideal. The Amiga should be fitted with a hard drive and running on Workbench 2.04 or higher. You should have at least 8MB FastRAM (preferably *much* more) and you will benefit

from a familiarity with AmigsDOS. The only 24-bit display card formally supported by LightWave on the Amiga platform is the Picasso by Village Tronic. Hard drive installation is completely automatic, the de-archiving and installer libraries are on *Disk 1*.

With Workbench running, insert *Disk 1*. Open *Disk 1*, double click the *InstallLightWave* icon and follow the instructions. I suggest you opt for the *Full Installation*. (You could simply install the *Layout* and *Modeler* modules, or the 'content' of *Objects* and *Fonts* archives). *Full Installation* is the best option, because you can't appreciate the programme fully without having everything on the drive. About 8MB of drive space will be consumed. Unfortunately, the silly 'Toaster' name survived the divorce from LightWave and will appear on any Amiga using the programme. Thus, the hard drive Installer will create a parent directory called *Toaster* and place everything in there, including a sub-directory called *3D*. The *3D* directory contains several important sub-directories, including *Objects*, *Surfaces*, *Scenes*, etc. *Toaster* will also contain a fonts directory called *ToasterFonts*, a *Utilities* directory, a directory for storing *Arexx* scripts, and the *tio* directory containing file-type converters. The main directory (*Toaster*) can be located wherever you want it on the drive. The required assigns are automatically written into the *sys:s/user-startup* script and a full installation will take about ten minutes.

By default, LightWave assumes that your hard drive partitions are labelled HD0: and HD1: If yours are labelled differently (e.g. DH0: and DH1:) several directories you will need to access will not automatically appear on file requesters. To reach them, you will have to type in their full path on the requester. You could get over this by simply adding additional assigns to your *sys:s/user-startup* file as follows:

assign HD0**:** DH0**:**
assign HD1**:** DH1**:**

Note however, there are two preference editors called *LW-config* and *Mod-config*, which allows a certain degree of customisation. They will be found in the *3D* drawer. More on these later.

You may prefer the main directory to be called something more sensible, like 'LightWave'. However, please do not change it without also changing the *s/user-startup* assigns to the new name and rebooting. For example, if you want to change the name 'Toaster' to 'Myname', on a hard drive called 'Drive', then the *sys:s/user-startup* assigns need amending as follows:

assign Toaster**:** Drive**:**Myname
assign 3D**:** Drive**:**Myname/3D

After software installation, you must reboot before running the programme. Then, when you open the *Toaster* drawer, you will be presented with two *LightWave* and two *Modeler* icons. If double-clicking the *Modeler* icon, or attempting to open it from *Layout*, exits the programme, you must turn off your computer and insert the security dongle into the Parallel (printer) port as described earlier. Then, after the reboot, everything should work fine.

LightWave's *Macro* functions require *Arexx* to be running. If it is not and you enter Modeler, you will get a message that *RexxMast* is missing. To fix this, just drag *RexxMast* (kept in the *sys:system* drawer) into *WBstartup*, or invoke it via the *s/user-startup* script. Either way, it will always be available when you boot.

LW-config* and *Mod-config

When you have become familiar with the operation of the software, you may wish to modify the default volume names used by file requesters to suite your hardware system. This and further, though limited customisation is possible by editing the contents of *LW-config* and *Mod-config*. These are ASCII text files and will be found in the *3D* drawer. Note that most of the text should be left alone or the programme may not work. It is particularly important not to alter the first two lines of either file. Other lines provide the default location for important directories like *Objects*. Here, you can adjust the pop-up file requesters to quote the names of any device connected to your system, e.g. replace the default HD0: and HD1: with DH0: and DH1:. Further, you might want to replace the DF1: drive with the label on your CD-ROM or Zip drive.

Four drives can be nominated. This is done by entering the *LW-config* file using a text editor, such as *Ed* (via the *Shell*). Here, you will see four lines beginning *FileReqPreset*. Each of these may be customised as DF0: , DH0: , CD0: , Zip0: , etc. Save the modified text and reboot. When file requesters next pop up, you will see the volume buttons bearing their new names.

The text contained in *Mod-config* affects the operation of the *Modeler* interface and allows some customisation. For example, you may prefer to have the supplied fonts (*PostScript* type 1 only) located in your Workbench (*Sys:Fonts*) directory. If so, you can relocate the *PSfonts* drawer into *Sys:Fonts* and amend the text to give the correct path.

Screen Mode

The *Modeler* is actually an independent programme that may be run in a variety of screen modes. These can be selected using the *Changemode* utility. If you look in the *Utilities* drawer you will see the *Changemode* tool and its documentation. Double-clicking *Changemode* pops a file requester asking you to load *Mod-config*. Load the file and you'll get a *Screen Mode* listing, which you can edit. Simply insert the reference number of the *Mode* in which you prefer *Modeler* to run and reboot. The selected *Mode* will be used whenever you enter Modeler directly from Workbench. However, if you enter it via Layout, the selected *Mode* will not be used unless *Mode Promotion* is active.

To enable any selected *Mode* to be used by both Layout and Modeler, go to the Workbench *Prefs* directory and open the *IControl* drawer. Here, you will see a button marked *Mode Promotion*. Click on this with the LMB to switch it on (i.e. highlighted). After a reboot, the selected *Sceen Mode* will be used for both Layout and Modeler.

Preamble

There are only two interfaces in LightWave, but they are very powerful and contain lots of menus, commands and sub-menus. One is *Modeler*, (American spelling!) which is used for constructing or modifying 3D objects. The second is *Layout*, which is used for arranging the objects, the lights, the camera and everything else together into a scene or animation. Layout also organises object surfacing, object movement, lighting effects, camera effects, frame rendering, frame and animation file saving, etc. The interfaces may be accessed and controlled using the mouse or by using keyboard equivalents. Mouse activity progresses by clicking with the cursor on various buttons. Each interface is button controlled, as are their sub-editors which regularly pop up. Pressing the *Help* key at any time will present you with a *Keyboard Shortcuts* panel showing all the keyboard equivalents for the current screen or editor. The following text assumes mouse operation unless otherwise specified. On-screen buttons are 'pressed' by placing the cursor on one and single-clicking with the left mouse button (LMB). Most operations involve the LMB, though additional functions may be available through the right mouse button (RMB). These are noted as they occur.

When you open the *Toaster* drawer, you will see *LightWave* and *Modeler* icons in both FPU and non-FPU versions (i.e. four icons). If you have a '060 processor, or one provided with floating point functionality, it would be advisable to delete the non-FP icons and avoid any complications mentioned in the *Changes.Txt* file. This appears immediately after finishing the installation, though you can read it any time in the *Toaster* drawer. If you don't have an FPU, then you'd delete the FP-icons. Note that some of the procedural textures (*Surfaces*) will not render well without FPU support.

The best way to initiate the programme is to double-click the *LightWave* icon with the LMB. This loads the programme and places you in the *Layout* environment. The *Modeler* environment can then be accessed from *Layout.* If you begin by entering *Modeler* first, you will not get direct access to *Layout.* You have to *Quit* to Workbench and then double-click the *LightWave* icon. Once *Layout* has loaded, you can switch between *Layout* and *Modeler* using the button at the top right corner of the screen. These are the only environments needed for the whole show. Before we raise the curtain however, there are several preliminaries you need to understand about *Points*, *Polygons*, *Curves, Objects* , *Surfaces* and other elements of LightWave theory. You must also be happy visualising things in three dimensional space. This is vital to the construction of an *Object*, the creation of a LightWave *Scene* and the production of a 3D animation. These and other important terms are discussed below. Each command and tool referred to in this preamble is described in more detail later.

Points

We use graph paper to plot the location of points on a two dimensional plane. The graph has two *axes* at right angles. These are conveniently labelled the X axis (left and right) and the Y axis, which is up and down. The position of a point on the graph paper is thus defined by its (X,Y) coordinates. When we are concerned with three dimensional space, a third axis, the Z axis, provides us with the 'front to back' element. A LightWave *Point* is a location in 3D space, defined by its (X, Y, Z) coordinates. A *Point* is used as a construction aid in modelling. Since *Points* do not have any dimensions, they cannot be 'seen' or rendered. However, a *Point* can be 'converted' by Modeler into a special type of *Polygon* known as a *Single Point Polygon*. (We'll talk about *Polygons* in a moment). Using Layout, a *Single Point Polygon* may be assigned colour and luminosity, characteristics normally applicable to extended *Surfaces*. *Single Point Polygons* are often referred to as *Particles*. Using *Particles* you can produce a star field, a shower of raindrops, etc. Animations involving the movement of *Particles* across the screen should be given motion blur for added realism. *Single Point Polygons* remain as points no matter how closely they are viewed or magnified. You can see constructional *Points* on the Modeler screen because they are important to object modelling and manipulation. They also remain as *Points* no matter how closely you view the Modeler screen. Here, a *Point* can be seen in two ways, *selected* and *unselected*. An overview of item selection is given later. *Points* are used as anchors to create *Polygons* and occur at every corner or *vertex*. You can also load *Points* into *Layout* and use them to influence *Objects* in a variety of ways, for example as a centre of rotation for an *Object*, *Light*, etc.

Polygons

The *Polygon* is the basic building block for LightWave *Objects*. *Polygons*, more specifically known as *Face Polygons*, are geometric shapes produced from *Points* and the lines which join them together. *Polygons* can have any number of sides, or perhaps a better word is edges, from zero upwards. A zero-edged *Polygon* is simply a *Point* (a *Single Point Polygon*). A single-edged *Polygon* is a straight line, produced by joining two *Points* in 3D space. *Single Point Polygons* and single-edged *Polygons* can be rendered as LightWave *Objects*, but most of the *Objects* you will use are constructed from multi-edged *Polygons* because these provide an extended *Surface*. LightWave allows you to construct *Objects* by joining *Polygons* together along common edges or at common points. The resulting *Object* resembles a sculpture made from wire netting and is called a wireframe. The 'wire' forming the *Object's* shape is sometimes referred to as the 'mesh'. LightWave *Objects* usually contain three- or four-sided (edged) polygons.

A three-sided (edged) *Polygon* (triangle) is always planar (perfectly flat), no matter what the location (coordinates) of its *Points*. The location of a triangular surface in 3D space is unambiguous and *LightWave* will always understand what it is handling. A triple-point polygon will always provide a *Surface* which *LightWave* can render successfully. However, a four-point or higher multi-point polygon may not be flat. It could look like a piece of writing paper folded across a diagonal, but with no edge there (if there was, we'd have two triangles!). Thus, non-planar polygons have ambiguous surfaces and may introduce rendering problems. This is particularly relevant in animations, where surfaces are constantly changing position.

You can ensure that any *Object* is constructed from triangles by pressing the *Triple* button in *Polygon* menu (more on the *Menus* later). The *Triple* algorithm examines the mesh and splits any four- or greater-edged *Polygons* into triangles. Additionally, if the triangles in the mesh (or part of the mesh) are too large, they may appear too flat when rendered. *LightWave* has some remarkable surfacing effects, but it cannot make a large flat triangle look like a ping-pong ball! The solution to over-large triangles will be found in the *Subdiv* (Sub-Divide) tool, also in the *Polygon* menu. *Subdiv* will only work with triangular polygons. The algorithm sub-divides a selected triangle into a smaller set of triangles. This treatment can be repeated more than once, until you're happy with the size of the units.

Curves, Splines, Knots and Patches

A *Curve* is the shape produced when two or more *Points* in 3D space are smoothly connected by a line. The path for the line is computed via a mathematical formula known as a *Spline*. Generally, the points along a *Spline Curve* are called *Knots*. *Curves* are modelling tools that enable you to create realistic *Objects* and can be *Saved* like *Objects* for later use, either in Modeler or Layout. *Curves* aid in the modelling of smooth-profile objects, yet they do not contain faces *per se*. When a *Curve* is used in this way, a group of polygons is created which follow the shape of the *Curve*.

If you connect *Curves* into an approximate outline of a shape you wish to create, you can generate smooth-surfaced *Patches* of polygons. These surface *Patches* can then be used to make free-flowing meshes of any shape and form. The creation of objects such as curtains, sails, car panels and many more shapes is a simple process using Modeler's *Spline Curves* and surface *Patches*.

Make a *Curve* by placing two or more *Points* in an *Edit Window*. Select them in order and click *Make* in the *Curves* group under the *Tools* menu. You can create a *Closed Curve* by connecting the *Start* and *End Points* together. The density of *Polygons* in a mesh created from *Curves* is determined by their *Curve Subdivision Setting* (found under *Objects*/*Options*). *Curve* subdivision is adaptive, meaning that the proximity of *Points* and the tightness of the *Curve* may cause the creation of more segments at that location . On the other hand, *Points* that are further apart will need fewer segments. Where appropriate, *Modeler* will add a few extra *Points* to maintain smoothness.

A *Curve* is a special type of *Polygon* (specifically called a *Curve Polygon)*, but which is not rendered like a normal polygon. *Curves* can be selected like normal polygons using the *Polygon* selection tool and then acted upon like other polygons, using the *Modify, Multiply, Combine* and *Polygon* menus. All *Curves* have a *Starting Point* and an *End Point*, according to the first and last *Point* selected prior to *Making* it. The *Start* is tagged with a white circle. Think of the *Curve* as a pathway, with a start and end. The *Start* and *End* can be reversed using the *Flip* command. In fact, the orientation of the *Starting* point (at one end or the other) affects the direction of the surface *Normals* in the mesh created from *Curve* manipulation. The ability to *Flip* the curve's direction is important, since it affects the direction that the resulting *Surface* will face.

Surfaces

A *Surface* is the area of space enclosed by the edges of *Polygons*. You can only 'see' an object as a solid entity if its *Polygons* have a *Surface* to render. (You can, however, render an *Object* as its *Wireframe* using a special switch in the *Surfaces* menu, found in Layout). LightWave allows any *Polygon* be *single-sided* or *double-sided*, meaning that it can have either one, or two, *Surfaces*. Thus, *single-sided Polygons* are visible from one side, but invisible from the other side. *Double-sided Polygons* are visible from either side. You can control in which

direction a *single-sided Polygon* faces. Normally, it should face 'outwards' towards the viewer (*Camera*), so that the *Surface* attributes will be visible when *LightWave* renders the *Polygon.* If the *Surface* is facing the wrong way, then it becomes invisible when rendered. It will then look like a hole in the rendered *Object*, though this could be what you actually wish to see in the render.

The direction in which a *single-sided Polygon* faces is indicated by *Modeler* whenever the *Polygon* is *Selected* (highlighted yellow) using the mouse, or other selection device. All selected *Polygons* have a perpendicular broken line (the *Normal*) emanating from the centre of its *Surface*. The *Normal* always extends from the viewable surface of the Polygon. The *Normal* should therefore point in a direction from which the surface could be seen by moving the *Camera*. *Polygons* facing the 'wrong' direction can be flipped 180° using the *Flip* command (*Polygon* menu).

The *Surface* of an *Object* usually consists of many joined-up *Polygons* for which you have assigned all individual sub-surfaces the same attributes. A *Surface* can be individually named and given attributes such as *Colour*, *Reflectivity*, *Transparency*, *Luminescence*, etc. These attributes are assigned via the *Layout* interface, but the allocation of names to specific parts of the mesh is done via *Modeler*. Any selected part of the mesh can be named using the *Surface* command (*Polygon* menu). *Surface* names are saved along with the *Object*. After named *Surfaces* have been allocated attributes in *Layout*, these too are saved with the *Object*. Any *Surface* not given a specific name will be called '*Default* '. All *Default Surfaces* render in pale grey until you give them a specific name and attributes. Many LightWave *Objects* are available from external sources with surface names and attributes already built in. Some are included in the *3D:Objects* directory.

Bones and Skeletons

You might think of a *Bone* as a tool, which you can use to displace large groups of *Polygons* in a mesh. Imagine that this *Bone* tool has a vector-like character, a 'force field' which allows you to influence the mesh according to the way you move it. Turn the *Bone* and a section of the mesh turns with it. Twist the *Bone* and the mesh will follow suit. The *Bone*/mesh combination can be likened to a real bone surrounded by flesh, where the mesh represents the skin. Inserting a LightWave *Bone* into an *Object* is like adding a powerful frame around which the *Object* is secured and supported. LightWave *Bones* can be made to influence only a small part of the mesh, or a large section of it, just like the bones in your body. So, adding *Bones* is adding realism to *Objects* that you want to bend and twist, just like a living thing. By inserting a more complex structure of *Bones*, you can get finer and finer control of the mesh. This is like making a *Skeleton* to fit the shape of the *Object* and any other 'extraneous bits' you wish to control. In fact that's what LightWave *Bones* create, a mesh-controlling *Skeleton*.

Metamorphosis

A *Metamorphosis* is the changing of one *Object* into another, like a caterpillar changing into a butterfly. LightWave has the ability to animate such a change using a process called *Metamorph*. It's often easier and perfectly acceptable to refer to *Metamorph* as '*Morph*'. When an *Object*, let's call it the *Starting Object*, is transformed into another, the new form is called the *Metamorph Target*. It's the *Target* for LightWave's *Metamorph* process. LightWave brings about the change by comparing the location of all the *Points* in the *Starting Object* with those in the *Metamorph Target*. To get things right, each *Point* in the *Starter* should have a corresponding *Point* in the *Target*. In other words, the *Starting Object* and the *Metamorph Target* should have the same number of *Points*. If they do not, the transformation cannot take place. In fact LightWave will complain if you try to *Render* a *Scene* containing a *Morph Target* with the wrong *Point* count. *Metamorph* animations are therefore generated using a *Starter* and a *Morph Target* created from the *same Object*. Two different *Objects* with an equal number of *Points* will not *Morph* properly. You will simply get a muddled mess of *Polygons*. LightWave needs to identify each *'Starter' Point* with a corresponding '*Target' Point* to create a satisfactory *Morph*. The process uses the *Point* coordinates of the *Morph Target*, so that LightWave can move the corresponding *Points* in the *Starter Object* to the new coordinates. Therefore, the *Morph Target* must be *Loaded* into the *Scene*, but it need not be visible at any stage.

A *Starting Object* may undergo a series of *Morphs* during the animation. This is done by creating a chain in which the first *Target* is the *Starter* for the next and so on. In this way, the *Starting Object* may have up to forty (40) *Morph Targets* in series. Furthermore, any number of *Starting Objects* in a *Scene* may undergo a *Morphing* operation. This illustrates LightWave's powerful data handling capabilities.

The extent of any *Morph* is determined by the *Metamorph Level.* A complete transformation is obtained when the *Morph Level* is set to 100%. However, you can *Morph* a *Starter* into its *Target* to any *Level* from 0.1 to 100%. When the *Starter* and *Target* have different *Surface* attributes, you can ensure that this difference is incorporated into the *Morphing* process. The ability to *Morph Surfaces* is an additional feature of the *Metamorph* routine.

The transformation from a *Starter* into a *Morph Target* will take place over the number of *Frames* in the *Scene* (*Animation*) unless you change that course of events using an *Envelope*. The *Envelope Editor* is discussed in detail later. An *Envelope* describes the change of an event over time. To enable a series of *Morphs* to take place

one after the other, a series of *Envelopes* is needed. These describe the transformation of each *Object* into the next at the appropriate stage of the *Scene* (*Animation*). As with all changes in LightWave's world, the incorporation of *Splines* is available for refining *Metamorph* transformations.

Lights

Light is crucial to the operation of LightWave's rendering system. As in the real world, you cannot take photographs or make a movie in total darkness. For this reason, LightWave insists that at least one *Default Light* source is present in a *Scene*. A *Scene* must also contain an all-pervading *Light* called *Ambient Light*. *Ambient Light* is analogus to diffuse daylight. It's always around to some extent, no matter how dim. You have control of its brightness so *Ambient Light* can put just the right amount of illumination into the darkest corners of the *Scene*. The *Default Light* and any additional *Light* sources must be one of three types, known as *Distant*, *Point* and *Spot*.

A *Distant Light* is similar to the direct light from the sun. It shines in one specific direction and illuminates the whole *Scene* from that direction. It is not focussed in any way. A *Point Light* is like a candle flame or a light bulb, it shines in all directions from the source. A *Spot Light* is the most regulated of the three types because it is directional and focussed. It only illuminates things which are within its cone of illumination. This *Light* behaves like a torch or a car headlamp.

The effect of any *Light* in a *Scene* is observed by the way *Objects* are illuminated by it. Their *Surface* colours are modified by the colour of the *Light* and shadows may be cast because of their relative positions. However, there will be occasions when the *Light* itself is 'visible' to the *Camera*, so what happens then? The simple answer is nothing! Any *Light* that is located within *Camera* shot will not be rendered. There will be no 'bright area' or 'bright spot' simply because the *Light* is 'seen' by the *Camera*. *Lights* provide illumination through the process of reflection, so LightWave renders their effects, but not their image. But what if you want a *Light* to be 'visible' in the render? The solution is the employ *Lens Flare*.

Lens Flare is an optical phenomenon created when light passes through a camera lens (or, indeed, the human eye). The lens medium causes the scattering and reflection of light rays, so that we observe the light's image as glare, often accompanied with a ring, a streak, a 'star'-like spiking, or some other optical distortion. *Flare* is also caused by the atmosphere which surrounds us. The stars in the night sky appear as twinkling 'stars' simply because of atmospheric distortions, plus those inherent to the human eye. Stars are actually point sources, much like LightWave's *Point Lights* and as we've noted, a *Point* is invisible because it has no dimension. So, if there were no optical abberation of starlight, we'd see very much less of the starry backdrop of night. Therefore, to render a *Point Light* or *Spot Light*, you need to apply *Lens Flare* to it. This is done by using appropriate settings in the *LightsEditor/Lens Flare* control panel. *Lens Flare* is usually avoided by photographers and movie makers, but it can be used to good effect and is essential if you want a *Light* to be 'visible'.

LightWave allows you to control all these *Lights* and their effects in very subtle ways, including their colour and brightness. You can control the way they cast shadows and how they dim with distance. You have control over other special effects such as the *Cone Angle* of a *Spot Light*, plus *Lens Flaring* and *Star Filtering*. All three *Light* types can be animated and their changes controlled through *Envelopes* and *Splines*. Each of these facilities is discussed in detail in the *Lights Editor* chapter.

Scenes

A *Scene* is LightWave's record of a defined arrangement of *Objects*, *Lights* and *Camera*. The *Surface* details of *Objects*, their position and orientation settings, their *Motion* paths, changes in lighting intensity, lens flare behaviour, *Image* utilisation, etc. are included. A *Scene* file also contains the numeral data used to render the images. The *First Frame*, *Last Frame* and *Frame Step* values are include in the file, which is *Saved* to the hard drive. *Scene* files are kept in the *3D:Scenes* directory. The characteristics of a typical *Scene* are described in the *Tutorial* section.

Key Frames

Key Frames are the crux of animating with LightWave. Animations are created by rendering a series of pictures, or *Frames*, in which one or more items (*Objects*, *Lights*, *Surfaces*, etc.) change from one *Frame* to the next. When these frames are displayed in rapid succession (up to 30 frames/second), an illusion of movement is created. Classical animations were made by hand-drawing each and every *Frame* (sometimes called a *Cel*) until the required number were produced. They were then photographed onto movie film. These films often needed tens, or even hundreds of thousands of drawings. Computer rendering software has taken all this hard work away and replaced it with a convenient system known as *Key Framing*.

Animation remains a demanding art, but the tedium of drawing all those cels is completely removed. *Key Frames* are set up in Layout using the large number of *Objects*, *Lights*, *Surfaces*, etc. at your disposal. Each *Key* represents an important stage in the latent animation, a stage which defines the state of all the elements of the *Scene* at a particular instant, on a particular *Frame*. Several such *Key Frames* are created throughout the length of the animation. Again, this might consist of a mere thirty *Frames*, or maybe thirty-thousand. The

computer simply needs the data contained in the *Key Frames*. It will then compile and render the intermediate cels, according to the 'instructions' you build into the *Keys*. This between *Key* in-filling is known by animators as 'tweening' though the mathematical term is interpolation.

Key to *Key* interpolation is used by LightWave in several facets of its operation. An important feature of the LightWave software is its use of interpolating *Splines*. *Splines Paths* allow animated *Objects* to behave in a realistic way around each *Key Frame*, as if they really are affected by natural forces, by gravity, inertia, centrifugal force, etc.

Null Object

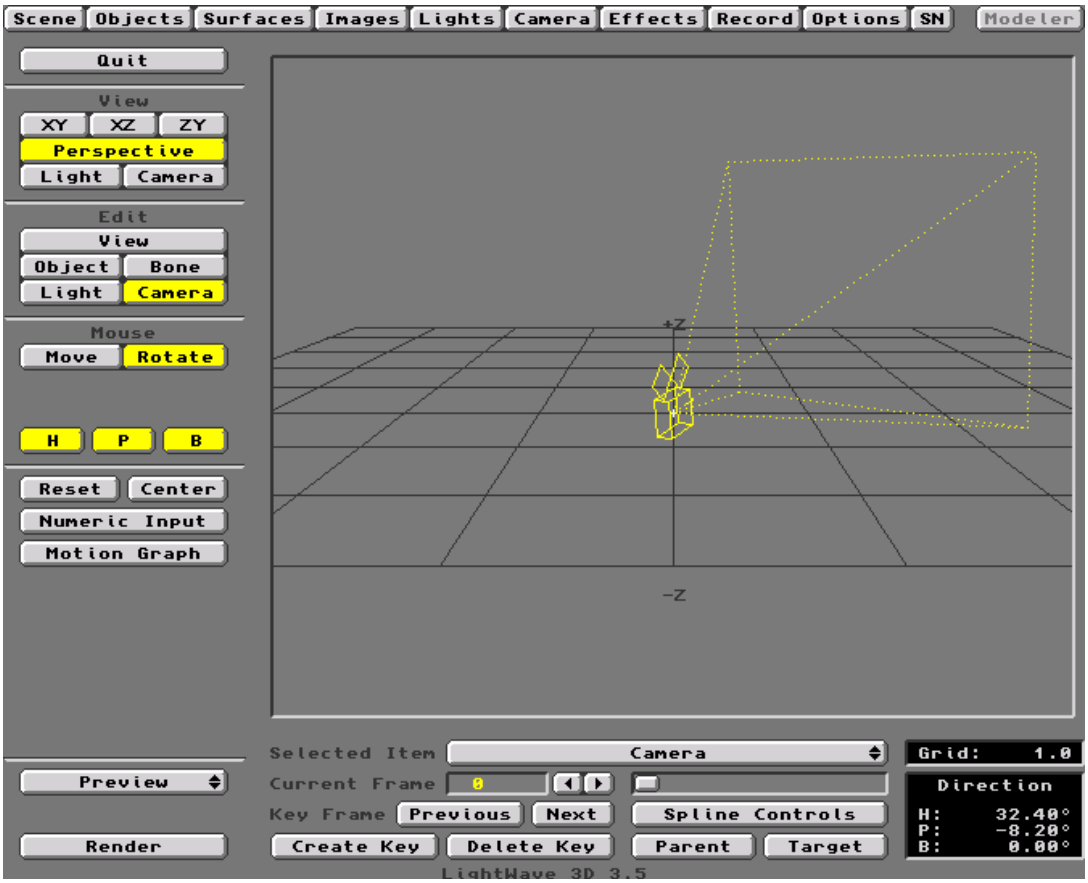
The *Null Object* is listed in LightWave's *Objects* directory as *NullObject*. It is a special kind of object because it contains no *Polygons*. In fact, the *Null* is the most basic object that LightWave understands. In simple terms, it is an axis in space. The *Null* cannot be rendered, but is displayed on the Layout screen as a six-pointed marker. You can *Select* it, *Move* it, *Rotate* it, *Size* it and *Stretch* it, but you cannot 'see' it in your productions. Therefore, you might ask what use has an *Object* which cannot be rendered? In fact, the *Null* has several applications, which enable you to produce even more impressive animations.

The primary use of the *Null* is as a *Parent* for other *Objects* in the *Scene*. The *Parenting* operation is discussed in more detail in the section describing the Layout Interface.

The *NullObject* is stored by itself in the *3D:Objects* directory. All the other *Objects* provided with LightWave are kept in various sub-directories, for example, in the *3D:Objects/Animals*, or the *3D:Objects/Aviation* directories. However, the *NullObject* is unique and even has its own load button. You'll find this on the *Objects Editor*, discussed in detail later.

The official LightWave manuals make very little reference to the *Null*, in spite of its prominance on the *Objects Editor* panel. Thankfully, *WaveGuide* puts this right and several useful applications for the *Null* are explored in *Tutorial 10.*

The Layout Interface



On entering the Layout environment, the large window provides you with a panoramic view of the 3D universe in which LightWave *Scenes* are created. Laid out before you is a horizontal *Grid*, bearing a *+Z* and *-Z* notation on the back and front edges. This *Grid* is a reference plane which guides you when positioning *Objects*, *Lights*, etc. in LightWave's universe. It is helpful to think of this area as a film set, which extends infinitely in all directions, including up and down. The visible *Grid* is simply the *XZ* plane of the *X*,*Y*, *Z* coordinate system. To the left of the *Grid* 's centre is the *-X* direction. To the right is *+X*. The *Y* axis (up and down) is omitted from the view to minimise clutter. In the centre of the *Grid*, you'll see a small rectangular object. This is the *Camera*, through which LightWave 'sees' the worlds you will create. Initially, the *Camera* looks backwards along the *Z* axis of the *Grid*, so you are seeing it from behind. That's why it doesn't look much like a camera when you first see it. We'll soon fix that, when you learn how to control everything you see. In the graphic above, the *Camera* has been rotated to the right. Using Layout's tools and commands, you are the film director, controlling the 'actors', their surface appearance, their interactions and movement. You will also control the cameraman, lighting, scenery and special effects. You'll create things you never imagined!

The row of buttons across the top of the interface control the resources you will employ to produce the desired result, three-dimensional, rendered images. These buttons are Layout *Menus* and are described in detail later. The separate button at upper right of the screen is labelled *Modeler*. Click on this to switch to the Modeler interface. When you do so, Layout will still be available through an analogous button on the Modeler screen. You can switch and interact between Layout and Modeler, thanks to the Amiga's multitasking capabilities. The button groups down the left of the screen will be described first.

Quit

Quit is self-explanatory, but when you exit LightWave after a *Scene* arranging or rendering session, various parameters are saved to the hard drive and are recalled the next time you *Load* that particular *Scene*. After clicking on the *Quit* button, you are returned to *Workbench*.

The groups of buttons below the *Quit* button are inter-related. Mouse manipulation of items in the *Scene* is controlled by the combined settings of buttons in *View*, *Edit* and *Mouse* groups. The options available in the *Mouse* group alter according to the type of item selected under the *Edit* group.

The View Group

This group contains six buttons with which you select the position from which you, as director, may view the stage. You are able to move around the stage using one or other of these buttons and gaze upon the *Scene* independently of the position of the actors (*Objects*), *Lights* or the *Camera*. When you first open the Layout interface, the default *View* position is *Perspective* as indicated by its button, highlighted yellow.

Perspective

Perspective is an overview, a useful vantage point from which you can see all elements of the *Scene*. You can see a *Grid* representing the XZ plane in 3D space. The *Grid* shows the X axis from left to right and the Z axis from 'front' to 'back' of the stage. The Y axis is 'up' and 'down', though to avoid clutter, it is not visible. This is a bird's eye view of the stage, the film set, or 'ground zero' if you like, looking down at an angle of about ten degrees. The stage is not solid, however, so items can take part in the *Scene* at any location above or below the XZ plane. In the centre of the *Grid* is a small rectangular object. This is LightWave's *Camera,* seen from behind. The *Camera* is used to take 'photographs' of the *Scene*. Animations are created by taking a series of photographs as elements of the *Scene* are changed over time. As in photography and film production, all your 3D renders are generated using the *Camera's* view.

XY XZ ZY

These are the orthographic view buttons. Click in turn on the *XY*, *XZ* and *ZY* buttons and both the *Grid* and the *Camera* will be seen from positions represented by the *XY* (Front), *XZ* (Top) and *ZY* (Right) positions. When you are setting up the various *Objects* in a LightWave scene, these views provide valuable insights for getting everything in the correct position.

Light

This provides you with a view of the set-up from the position of any *Light* that you place in the *Scene*. The view from a *Light* can be helpful in seeing which parts of the *Scene* are illuminated, especially by spotlights. You select the *Light* you need using the *Edit* group (below).

Camera

This is the most important view of all because it looks through the *Camera's* viewfinder. This is what you will get (from the positional aspect) when the *Scene* is rendered.

The Edit Group

Clicking on one or other of the five buttons in the *Edit* group allows you to select the item or item type you wish to manipulate (edit). Thus, you can make the director's *View*, an *Object*, a *Bone* within an object, a *Light* or the *Camera* your *Edit* item. Depending upon which button is clicked, the *Mouse* menu will change to reflect the properties of the *Edit* item. Further, if the *Object*, *Bone*, *Light* or *Camera* button is clicked, an additional set of five buttons appears at the bottom edge of the Layout screen. Four of these buttons enable you to manipulate individual *Frames* within a series used for an animation. The fifth button allows you to associate the movement of one or more *Objects* in the animation with that of another, *Parent Object*.

View

View allows you to manipulate the director's overview or vantage point according to what is active under the *View Menu* (i.e*. XY, XZ, ZY or Perspective*).

Object

The *Object* button allows you to make any *Object* loaded into the *Scene* the edit item. If there are several *Objects* to choose from, use the *Selected Item* scroll bar under the main window to find and select the one you want. Click on the bar with the LMB and a list pops up. Move the cursor to the relevant *Object* and release the LMB. The required *Object* will be highlighted yellow in the main window. Clicking the *Object* button brings several new buttons into play. These are described below.

Bone

This button selects a *Bone* (if present) as the *Edit* item. A *Bone* is an optional accessory in *Object* animation. If used, it forms part of the *Object Skeleton*. *Bones* are used for creating realistic deformations within an *Object*.

By creating a skeleton-like framework within the *Object*, you are able to control movement of the *Object's* mesh (see Modeler for details of the mesh). *Bones* are individually named to identify them. Select the required *Bone* using the *Selected Item* scroll bar. It will appear highlighted yellow in the main window (see the *Preamble*, *Object Menu/ Skeleton* and *Bones* below).

Light

This button selects a *Light* as the edit item. *Lights* are individually named to identify them. If several *Lights* are present, choose the required one using the *Selected Item* scroll bar. The selected *Light* will be highlighted yellow in the main window. If *Light* is also selected in the *View* menu, the window will show the view from the *Light* as it is edited. Selecting a *Light* causes a new buttons (*Spline Controls*, *Parent* and *Target*) to appear at the bottom of the Layout screen. These are described below.

Camera

Selects the *Camera* as the edit item. A view of the *Camera*, or the view through the *Camera*, will be shown in the main window according to the setting in the *View* group. In any other mode than *View/Camera*, the *Edit/Camera* function will highlight the *Camera* in yellow and will show its *Field of View* as a dotted line, otherwise, you will see what the *Camera* sees as you edit it. Selecting the *Camera* causes a new buttons (*Spline Controls*, *Parent* and *Target*) to appear at the bottom of the Layout screen. These are described below.

The Mouse Group

The *Mouse* group provides several options for the use of the mouse when editing. With the *Edit* group in *View* mode, there are six *Mouse* options. With the *Edit* group set to *Object*, *Bone*, *Light* or *Camera*, there are eight *Mouse* options. Note that any changes made to the default position, orientation, etc. of an *Object*, a *Bone,* a *Light* or the *Camera* will not become permanent until you click the *Create Key* button. See also *Create Key* button below. To examine all the buttons in the *Mouse* menu you will need to select appropriate *Edit* buttons.

Move

Move allows you to use the LMB or the RMB to control linear movement of the edit item. The LMB allows motion in a horizontal plane (X axis and Z axis). The RMB allows vertical control along the Y axis. The coordinates of the edit item are indicated in the small window at bottom right. The actions available under *Move* are constrained according to the selection of the three buttons (***X***, ***Y*** and ***Z***) below the *Mouse* group. See also *Create Key* button below.

Rotate

Rotate changes the response of the edit item to rotational. The LMB permits control of *Heading* (*H*) and *Pitch* (*P*), while the RMB controls *Bank* (*B*). The orientation of the edit item is shown in the small window at bottom right. The actions available under *Rotate* are constrained according to the selection of the three buttons (***H***, ***P*** and ***B***) beneath the *Mouse* group. See also *Create Key* button below.

Size

Size allows rapid adjustment of the *Size* of any *Object* using the mouse. Select the required *Object* using the *Edit* group, then click the *Size* button and scale the *Object* with either the RMB or LMB. Note that changing an *Object's Size* in Layout only changes it within that particular *Scene.* It does not make any change to the *Object* stored in the *3D/Objects* directory. See also *Create Key* button below.

Stretch

Stretch is a special form of sizing with which you can constrain the change to one or more axes, or make different degrees of change to each axis. *Stretch* is constrained along the three axes according to the selection of the *X*, *Y* and *Z* control buttons. The LMB is used to *Stretch* in the *X* and *Z* directions and the RMB stretches in the *Y* direction. See also *Create Key* button below.

Move Pivot Pt

Move Pivot Point allows you to amend the axial *Origin,* which is stored with an *Object* file following its creation in Modeler. Each time an *Object* is loaded into Layout, a small 'x' is loaded with it. This is the *Origin* relative to the *Object* at its creation. In Layout this is called the *Pivot Point*. Any change made to an *Object's* orientation using the *Mouse/Rotate* options employs the *Pivot Point* as the centre of rotation. Thus, rotating an *Object* using the *Y* constraint usually causes it to spin 'on its axis' like an ice skater. However, you may wish the motion to be more orbital, like a skater going around in large circles. To do this efficiently you should *Move Pivot Point* to a location outside the 'base' of the *Object*. Select the *Object* as the *Edit* item and click the *Move Pivot Point* button.

Providing no changes have been made to the object's *Size* or *Rotation*, the 'x' can now be edited with the LMB or RMB in the same manner as *Move*. The new *Pivot Point* is fixed as soon as you stop editing. It does not need to be saved in any way. Note that the new *Pivot Point* is stored within the *Scene* script and makes no changes to the *Object* file stored in the *3D:Objects* directory. If changes to the object's default *Size* and *Rotation* have been made within the *Scene*, a warning message will pop up when you press the *Move Pivot Pt* button. This tells you to reset the *Object's* scale and rotation before moving the *Pivot Point.*
All parameters of a *Size* and *Rotation* change must be reset using the *Reset* button, located just below the *Mouse* menu. Ensure that you *Reset Size* with the *X*, *Y* and *Z* buttons selected and *Reset Rotation* with the *H*, *P* and *B* buttons selected. This will cause the *Object* to jump back to its default state (or the state following the last *Move Pivot Point*). You can then edit the *Pivot Point* as described above. The location of the *Pivot Point* can also be *Reset*. You may find it less trouble to anticipate the need to *Move Pivot Point* and do this immediately after loading the default *Object*.

Rest Length

This button invokes a special form of *Stretch* associated with *Bones*. The *Rest Length* of a *Bone* determines its size in relation to the *Object* it is assigned to. *Bones* are loaded into Layout with a default size of 1 unit of distance (1 metre). This is often too large to be used within the *Skeleton* of an *Object*. To scale the *Bone* to the *Object*, you must use the *Rest Length* button to amend the physical size. If you do this simply using the *Size* button, the resized *Bone* will behave as if it were still 1 metre long (see *Bones and the Object Skeleton Panel* below).

Reset

Reset allows you to return any changes made in the position, scale and orientation of the *Edit* item back to their as-loaded, default states. This is an important button for the *Move Pivot Point* tool (see above).

Centre

Centre allows you to adjust the overview position so that the selected item appears in the centre of the main window. The item is not moved, only your viewing position. This is useful for editing items which have become displaced from the central area of the stage. *Centre* works in conjunction with the *XY*, *XZ*, *ZY* and *Perspective* options in the *View* menu. The location of an *Object's* centre is the *Pivot Point* so this will be centralised in the window when you invoke the *Centre* command. If the selected item is displaced well away from the visible *Grid*, the *Centre* command can cause you to lose track of your location. This is especially true when your viewing position is providing 'close up' images. If you lose your whereabouts, switch to *Edit/View* and *Zoom* out, either with the mouse of via a *Numeric Input*.

Numeric Input

Allows precise numeric data to be entered into the pop up panels which appear whenever you click the button. This button operates in conjunction with the five editing buttons under the *Mouse* menu. This button is most useful for making small and accurate changes to a positional or scaling parameter after getting things about right using the mouse.

Motion Graph

If you select an *Object*, a *Bone*, a *Light*, or the *Camera* and you then click on the *Motion Graph* button, the *Motion Control Editor* pops up on top of the Layout screen. If a suitable item has not been pre-selected, an error message will appear. A *Motion Graph* (or *Motion Path*) is a graphical representation of the position or orientation of an *Object*, a *Bone*, a *Light* or the *Camera* as it changes over time, i.e. from *Frame* to *Frame*. These graphs are 'plotted' automatically whenever you make a change the location and orientation parameters between *Key Frames* (see *Create Key* and *Delete Key* below). When you *Save* the resulting *Scene*, LightWave stores these data within the *Scene* file, so that they are recalled each time the *Scene* is loaded. A detailed account of the *Motion Graph Editor* is given below.

Beneath the ***Selected Item*** scroll bar is another set of buttons which allow you to access different *Frames* of an animation.

Current Frame

The *Current Frame* facility allows you to access any frame of the animation loaded into Layout and make it the *Current Frame*. This can be done in three ways. Using the keyboard, you can enter the number of the frame you wish to access by over-typing the existing figure and pressing the Return key. Or, you can click either of the *Frame Step* buttons (**<**) and (**>**) to move one frame at a time forwards or backwards. Or, you can 'zoom' to the required frame using the LMB on the slide bar. The *Current Frame* number will change accordingly.

Key Frame* / *Previous* / *Next

The *Key Frame* buttons allow you to skip between *Key Frames* of the current animation. Whatever the *Current Frame* number, clicking on *Previous* or *Next* will immediately transport the view to the previous or the next *Key*

Frame. *Key Frames* throughout the animation can be located quickly by repeat clicking on the appropriate button.

Spline Controls

Each *Key Frame* has an associated set of *Spline* data. Whenever the *Current Frame* is a *Key Frame*, the *Spline Controls* button is displayed. *Splines* are mathematical *Curves* used to control the shape of a *Surface*, an *Object's* rate of change in velocity (acceleration and deceleration), the rate at which a *Light* 's intensity may rise or fall, etc. The shape of a *Spline Curve* is dictated by a number of *Points* or *Knots* along its length. There are few *Points* where the shape of the *Curve* is linear and many *Points* where the shape of the *Curve* is acute. The mathematical data (the rate of change) contained in a *Spline* is therefore expressed by the position of *Points* along the *Curve*. *Splines* are used in LightWave's Modeler to generate smooth organic shapes. They are also discussed in the Preamble.

Layout actually generates a *Spline* or *Motion Path* whenever the location of an item changes between one *Key Frame* and the next. This *Path* automatically alters shape as more *Keys* are added to the chain. If the *Show Motion Paths* option is active (*Options* menu), it is displayed on the main window. A *Motion Path* is drawn as a white line, each end of which bears a small cross (+). This indicates the location of the item's axis at each *Key Frame*. As *Keys* are added, these index marks appear along the length of the *Path*. Between the index crosses are *Frame* marker dots, which are analogous to the *Knots* described above. Each *Knot* is the location of the item's axis for the intervening *Frames*. To avoid the display becoming cluttered, only the *Spline Path* of the currently selected item is shown. This *Spline* may become partially hidden during the manipulation of the item. Using the *Spline* principle in Layout, you can control the rate of change of any item parameter (speed, rotation, brightness, etc.) either side of a *Key Frame*. This allows for much more realism (and special effects) to be achieved in your animations. A simple, but good example is a bouncing ball animation.

Digression..... The Bouncing Ball Animation (aka 'Boing')

Imagine that you are making an animation of a ball bouncing on the spot.
In its simplest form, this would consist of a series of *Frames* showing the ball alternately on the ground and in the air. Clearly, this wouldn't be very lifelike, even if played at one frame per second. Each frame could be regarded as a *Key Frame*, though in this case they wouldn't help either. What is needed are more frames in between the *Key Frames,* showing the ball in intermediate positions. We might therefore separate each pair of *Keys* with say, nine extra frames. LightWave would then interpolate each tenth of the way down and each tenth of the way up, to provide the intermediate frames and an animation looking somewhat more lifelike. However, this bouncing ball anim still lacks something. That touch of finesse to make it perfect.

If you observe a real bouncing ball, it accelerates to the ground, which brings its motion to an abrupt halt. The ball remains stationary for a brief moment and its kinetic energy is transferred mainly into potential energy, plus heat and sound. This provides the ball with the capacity to bounce back into upward motion. It accelerates upwards from its fleeting standstill. However, about half way up the bounce it begins to decelerate, until at the top it is stationary again. Here, its remaining kinetic energy has become potential energy once more. This process is repeated, each bounce achieving less and less height as the energy is dissipated as heat and sound.
To simulate this motion, we use *Splines*. *Splines* are needed at any point in the ball's path where a significant change in direction and velocity occurs. *Splines* are used because none of these changes are instantaneous. *Splines* allow accelerations and decelerations to be rendered smoothly without you having to calculate each frame individually.

But *Splines* can do more than simply 'damp' down velocity or directional changes. They can be used to accentuate behaviour at *Key Frames*. They can be used to convey an impression of 'anticipating' or even of 'over-shooting' the *Key*. *Splines* are not restricted to the movement of *Objects*. Any changing parameter can be modified using *Splines*. A change in *Light* intensity, a zoom-in, a fade-out, a change in the *Camera's* focal length, etc. may all be tailor-made for the job using *Splines*. These special effects are achieved through use of the *Tension*, *Continuity*, *Bias* and *Linear* options in LightWave's *Spline Control* system. The effects of *Splines* are visualised in the *Motion Graph*, whose button is always available on the Layout screen. In fact, you may find that manipulation of the *Motion Graph* is a more natural method of using *Splines* because of the immediacy and fluidity. *Splines* add a vital spark to animations that would be almost impossible to achieve by any other method.

Now back to ***Spline Controls*****.....**
One way to manage displacement *Splines* (*Spline Paths*) is through the *Spline Controls* button, which appears beneath the frame selection slide bar. Clicking *Spline Controls* pops up the *Current Key Frame Spline Controls* panel. The *Splines* presented here are associated with any selected item in the *Scene*. *Spline Path* data are numeric parameters which may be entered (keyboard) into the *Spline Controls* panel. There are three fields into

which a number can be entered. Valid settings for all three fields are between -1 and 1, with 0 being the default value. The fourth *Spline Control* parameter (*Linear*) has just an on/off option.

Spline Controls/Tension affects the apparent velocity of an *Object* (or other selected item) as it passes through the *Key Frame*. It does this by amending the position of the item over several frames either side of and including the *Key Frame*.
Positive *Tension* (0 to 1) causes a moving *Object*, etc. to slow down ('ease in') as it approaches the *Key* and then accelerate ('ease out') away from the *Key*. As the *Tension* setting approaches 1, the greater is the change in speed. A *Tension* of 1 will cause the *Object*, etc. to be stationary at the *Key*.
Without *Tension* (0), the *Object*, etc. will pass through the *Key* at constant velocity.
A *Negative Tension* setting (-1 to 0) will cause the *Object*, etc. to accelerate into the *Key Frame*. On passing through the *Key*, the *Object*, etc. will then decelerate to its former incoming velocity. The motion of an *Object* may also affected by a *Spline* from the next *Key* Frame if it lies especially close to the current *Key*.
In the Bouncing Ball anim above, you'd use a *Tension* of 1 for the *Key Frames* at the very top of each bounce. This will make the ball stop momentarily, before falling again. To simulate the ball slamming into the ground under acceleration, you might set a *Tension* of -1 for each *Key* containing a collision. *Tension* is usually used in conjunction with *Continuity* and *Bias* to achieve the best visual effect.

Spline Controls/Continuity affects the smoothness with which the *Key Frame* is incorporated into the *Spline Curve*. *Continuity* has no effect on events away from the *Key*. It adjusts the position of the *Object at* the *Key* only. It may be used to accentuate a change or a break in an *Object's* motion path.
With a *Continuity* setting of 0, the *Spline* will curve fluidly through the *Key*. This will result in the *Object* moving beyond a *Key Frame*, particularly one associated with a change of direction. The resulting *Spline Path* will take a smooth, natural course.
At a setting of -1, the *Spline Curve* will abruptly change direction at the *Key*. This makes the action 'stop in its tracks' then reverse. This is useful if you want an *Object* to make a sudden, almost jerky change of direction at the *Key*. In the Bouncing Ball anim above, the *Keys* showing the ball touching the ground will benefit from a *Continuity* setting of -1.
A positive setting of 1 will cause the *Spline Path* to over-compensate before and after the *Key Frame*, causing an unsatisfactory 'stutter' in the *Object's* motion. For this reason, positive *Continuity* settings are not often used.

Spline Controls/Bias can be used to off-set (*bias*) the curvature of the *Spline Curve* before or after the *Key Frame*. Positive *Bias* settings cause the *Object*, etc. to 'overshoot' the *Key* before the *Tension* or *Continuity* effects take place. Negative settings cause the *Spline* to 'anticipate' the *Key Frame* and exhibit the effects slightly early. A *Bias* of 0 means the effect of the other parameters will be unbiased around the *Key Frame*. *Bias* serves to accentuate certain types of motion, whereby incoming motion undershoots (anticipates) the *Key* or outgoing motion overshoots the *Key*. It's akin to shifting the *Key Frame* to a slightly earlier or slightly later position without actually doing so. *Negative Bias* is often used in cartoon character animation. It helps generate our anticipation of an event involving a rapid, but expected change in direction.

Spline Controls/Linear will cause the *Object*, etc. to move directly from the previous *Key* to the current *Key Frame* without experiencing any *Spline* deviation. With the *Linear* button selected, all other *Spline* parameters are voided between the selected *Key* and its predecessor. It does not, however, effect *Spline* behaviour between the selected *Key* and the following *Key*.
An *Object*, etc. moving towards a *Key Frame* assigned *Linear* (highlighted yellow) will do so in a straight, linear fashion. It will be unaffected by any numerical settings.

Note: The shape of the *Spline* in the Layout window will give a good impression of how the *Object*, etc. will behave during the animation. Changes in velocity can be equated to changes in the spacing of the *Knots* between *Key Frames*. As an *Object* decelerates, the *Knots* become closer together and vice versa. In the case of the Bouncing Ball, the *Spline Path* is superimposed on itself as the ball rises and falls. However, with *Tension* (-1) and *Continuity* (-1) set as recommended, the motion is very realistic. Compare this with a twenty *Frame* bounce from left to right. The *Spline Controls* for this are quite different because a second dimension has been introduced. These *Spline Control* differences are readily appreciated using the *Preview* facility (see below). In general, you should adjust *Splines* using the *Motion Graph* facility, which allows you to examine motion in all three dimensions (see below).

Create Key

The *Create Key* button is one of the most important of all. It fixes in place any *Object*, *Bone*, *Light* or *Camera* positions that you have set up by the use of the *Edit* and *Mouse* menus. When *Objects* and *Lights* are loaded into Layout, they will initially be positioned near the *Origin* of the *Grid*. This is also the *Camera's* default location when you first open the interface. Any change in position or orientation of any editable item must be fixed by clicking the *Create Key* button. Thus, the current *Frame* can be made a *Key Frame*, meaning that the data

related to the location and orientation of the item, or items, is stored in memory. The *Key Frame* number must be confirmed as follows.

When you click on *Create Key*, a pop-up panel indicates the current *Frame* number. You may edit this, if necessary, using the keyboard and pressing *Return*. The current *Frame*, or other nominated *Frame* will then be assigned the positional etc. data and made into a *Key Frame*. The specific items assigned to the *Key Frame* are determined as follows.

Create Key / Selected Item
Assigns only the selected item to that *Key Frame*.

Create Key / All Items
Assigns all items (*Objects*, *Bones, Lights* and *Camera*) to that Key *Frame*.

Create Key / Selected Item and Descendants
Assigns the selected item, plus any other which you have made its 'child' item. Child items are created using the *Parent* button (see below).

When you are happy with the selection, click *OK*, otherwise click *Cancel* to cancel the *Create Key* command.

Frame 0 is usually (though not necessarily) a *Key Frame*. If it is not, your animations might commence with all items near the *Origin*. Also see *Delete Key* below.

Remember that each item in the *Scene* can have its own individual *Key Frames*. Any *Key Frame* may be assigned with a different selection of items from the others. Make sure that you select the correct item (an *Object*, a *Light* or the *Camera*) before creating the *Key Frame*. Never make more *Key Frames* than are absolutely necessary!

Delete Key

This button will remove the *Key* status from any nominated *Key Frame*. You can skip between *Keys* using the *Previous* and *Next* buttons, before invoking the *Delete Key* command. Alternatively, click on *Delete Key* and type in the number of the *Key* you wish to *Delete* in the pop-up panel. There are three options for which *Key* status may be deleted.

Delete Key/Selected Item
The *Key* status of the selected item only will be deleted.

Delete Key/All Items
The *Key* status of all items will be deleted.

Delete Key/Selected Item and Descendants
The *Key* status of the selected item and its 'children' will be deleted. Child items are created using the *Parent* button (see below).

When you are happy with the selection, click *OK*, otherwise click *Cancel* to cancel the *Create Key* command.

NOTE: You cannot *Delete* the *Key* status of the *First Key Frame* in a series of *Keys*. *Frame 0* is usually a *Key Frame* and if so, its *Key* status will not be deleted. If *Frame 0* is not a *Key Frame*, then the *First Key Frame* cannot be deleted. Any attempt to do so will cause an *Error* message to pop up.

Parent

The *Parent* button allows you to make any *Object* into a *Parent*. The *Parenting* operation creates an association between any item in the *Scene* and that *Parent*. An item *Parented* by an *Object* will mimic the change in direction, rotation or speed of its *Parent*, whilst undergoing any changes defined within its own *Key Frames*. Thus, *Objects*, *Lights* and the *Camera* may be *Parented* by another *Object*. An item connected to a *Parent* in this way can be considered a *Child*. A *Parent* may have several *Child* items, but each *Child* can have but one *Parent*. A *Child* cannot *Parent* itself nor can it be assigned as its *Parent's Parent*. LightWave will detect this as an infinite parenting loop and remove it. Under such circumstances an *Error* message will confirm its action. The error will cause the looped *Objects* to superimpose and you must reposition them and create the *Parent* again. There may be several *Parents* in the *Scene*, provided there are enough available items to connect with them. If necessary, *Objects*, *Lights* and the *Camera* may be *Parented* by *Child Objects*, producing a hierarchy.

Select the item to be parented, click the *Parent* button and the *Parent Object* panel pops up. This contains a scroll bar containing the word (*none*). This indicates that the selected item currently has no *Parent*. Click on the scroll bar with the LMB and a list of all the *Objects* in the current *Scene* pops up. This list includes the selected

(latent *Child) Object*, but no action will result if this is selected. Move the cursor with the LMB to the desired *Parent* and release the LMB. The *Object's* name appears in the scroll bar. This is now the *Parent* of the selected item. Click on *OK* to confirm. *Cancel will* cancel the *Parenting* operation. The selected item (the *Child*) will now mimic its *Parent's* motion throughout the *Scene*.

Target

Target is a control associated with *Lights* and the *Camera* only. A nominated *Target Object* will remain the 'target' of the selected *Light*, especially when the *Light* is directional like the *Distant* and *Spot* forms. Each type of *Light* will move and/or rotate so that it always 'looks at' the *Target Object* , wherever the *Object* moves on the stage. An *Object* nominated as *Target* for the *Camera* will be kept within the 'viewfinder' of the *Camera* as the *Object* moves around the stage. With a *Light* or the *Camera* is selected as the *Edit* item, click the *Target* button. The *Target Object* panel pops up. This contains a scroll bar in which a *Target Object* can be nominated. The default is (*none*) indicating that there is no *Target* associated with the current *Edit* item. Click the scroll bar with the LMB and a list if all *Objects* in the current *Scene* pops up. With the LMB held down, select the required *Target Object* from the list. Release the LMB. The selected *Target* name now appears in the scroll bar. Click *OK* to confirm the selection. Click *Cancel* to cancel the *Target* operation. Clicking *OK* causes the selected *Light* or the *Camera* to orientate itself 'towards' the *Target* and will remain so throughout any subsequent motion of the *Target Object*.

Preview

The *Preview* button located in the bottom left corner provides access to Layout's animation preview system. LightWave can generate real-time wireframe previews of the current project. This is displayed in the Layout window. The animation is committed entirely to RAM and cannot be saved beyond the current session. This enables you to see how your animation project is developing, or to make a final check before committing the *Scene* to the full rendering process. The *Preview* button is a multi-choice button, which pops a list of three options when clicked on with the LMB. With the LMB held down, move the cursor through the list to the desired option. When this is under the cursor release the LMB.

Preview/Make Preview pops up the *Make Preview* panel. This option starts the wireframe rendering process. It contains three numeric input fields and two options.

> ***First Frame*** is the number of the *Frame* at which the *Preview* will start.
> The default is Frame 1, the usual starting frame for animations. Frame 0 is typically used as an index frame, which stores information about the project, the *Object* list, *Lights* list, etc. This is why Frame 0 ought to be a *Key Frame*. LightWave defaults to Frame 1 when rendering is invoked. However, neither of these defaults is fixed and you may begin the *Preview* from Frame 0, or any other *Frame* number. Type in the number via the keyboard and press *Return*.
>
> ***Last Frame*** is the number of the *Frame* at which the *Preview* will stop.
> The default in Frame 30, the same as invoked under the *Scene* menu (see below). Type in the appropriate *Last Frame* number for your *Preview* using the keyboard and press *Return*. The maximum length of the *Preview* is controlled by the total number of available *Frames*. The maximum number of Frames is also limited by the available RAM.
> RAM can be rationed by using a larger *Frame Step* (below).
>
> ***Frame Step*** is the step between the *Frame* numbers rendered in the *Preview*. The default *Frame Step* is 1, meaning that every frame in the range specified will be used. Enter a larger number to render the *Preview* in larger steps. For example, a value of 3 will cause every third *Frame* to be used until the *Last*, or nearest to *Last Frame* number is reached.
> See also *Frame Step* in the *Scene* menu below.
>
> ***Preview Type*** has two options:
>
> > ***Bounding Box*** will generate the *Preview* animation using *Object Bounding Boxe*s only. This provides for a very rapid render and is often sufficient for preliminary evaluation of *Object* paths, etc.
> > ***Wireframe*** will generate the *Preview* animation in wireframe format.
>
> When you are happy with the settings in the *Make Preview* panel, click *OK*, or click *Cancel* to cancel the *Preview* operation.

Clicking *OK* puts Layout into the *Preview* creation mode. The cursor will show 'busy' and the main window will display each *Preview Frame* as it is created. The number of the current frame within the animation is displayed in the bottom right corner. This ranges from 1 upwards, irrespective of the *Start Frame* number.

The *Preview* animation will be generated to the *Last Frame* unless you interrupt the process. Pressing the *Esc* key interrupts and halts the process. It cannot be resumed, though the incomplete animation is retained in RAM and may be displayed using the *Preview Playback Controls*, discussed below.

As soon as the *Preview* has generated, the Layout interface becomes ghosted (inoperable) except for the pop-up *Preview Playback Control* panel and the display window. This state is also invoked if the *Play Preview* option is selected via the *Preview* button and a *Preview* animation is residing in RAM.

Note: Any *Make Preview* invoked will be generated using the currently selected *View*. This means you can *Preview* the animation from a number of aspects. If you wish to *Preview* the render proper, you should ensure that the *View* group is set to *Camera*. The *Preview* will also be generated using current *Visibility* settings (see *Options* menu). The size of the *Preview* animation is limited by available RAM.

Render

Clicking on the *Render* button causes the *Render Scene* panel to pop up. Entering numeric data into this panel prepares LightWave to *Render* the current *Scene* in 24-bit. The rendering process may be carried out over a range of *Frames* to provide a series of individual image files or, the rendered frames can be compiled into an animation file or, any individual *Frame* can be rendered in isolation. The panel contains a summary of various settings which will be used. These are displayed in the black screen covering the lower half of the *Render Scene* panel. The upper half of the panel contains three numeric input fields and a *Frame Advance* selector consisting of two option buttons. Details of the panel are given below. The insertion point for numeric data can be made using the RMB. Note that the saving of images is controlled using the *Record Menu*.

First Frame This field defaults to 1, the normal starting *Frame* for LightWave renders. The *Starting Frame* number can be changed by entering a new number using the keyboard and pressing the *Return* key.

Last Frame This is the number of the last *Frame* of the animation to be rendered. Insert the desired *Frame* number using the keyboard and press *Return*.

Frame Step This is the step made by the rendering process in moving from one *Frame* to the next. This field defaults to 1, but can be set to any suitable value via the keyboard.

Frame Advance has two options. Select one by clicking on the appropriate button.

Frame Advance/Manual Select this option to render a single *Frame*. The *Frame* rendered will be according to *First Frame* setting.

Frame Advance/Automatic Select this option to render a sequence of images. You must have *RBG* image, *Alpha* image or *Anim* file saving turned *On*. The appropriate pathnames are also required when saving to the hard drive, single frame video recorder or other recording device. The rendering process will automatically advance as each *Frame* is rendered.

The information given in the black screen is set using the *Camera* and *Record* menus (see below). This provides an overview of how the rendering shall proceed, including what elements of the process will be saved and where they will be located on the hard drive.

Resolution: This is the pixel resolution of the rendered image.

Antialiasing: This tells you whether any antialiasing algorithm will be used during the render.

Render Display: Indicates how the rendered image will be displayed on the monitor screen as it is completed. In the absence of a graphics card, the options are 6-bit or 8-bit HAM.

Save ANIM: Indicates *Off* or *On*, according to whether anim file saving has been selected under the *Record* menu (see below).

Save RGB: Indicates *On* or *Off*, according to whether *RGB* image data saving has been selected under the *Record* menu (see below)

Save Alpha: Indicates *On* or *Off*, according to whether *Alpha* channel recording has been invoked under the *Record* menu (see below).

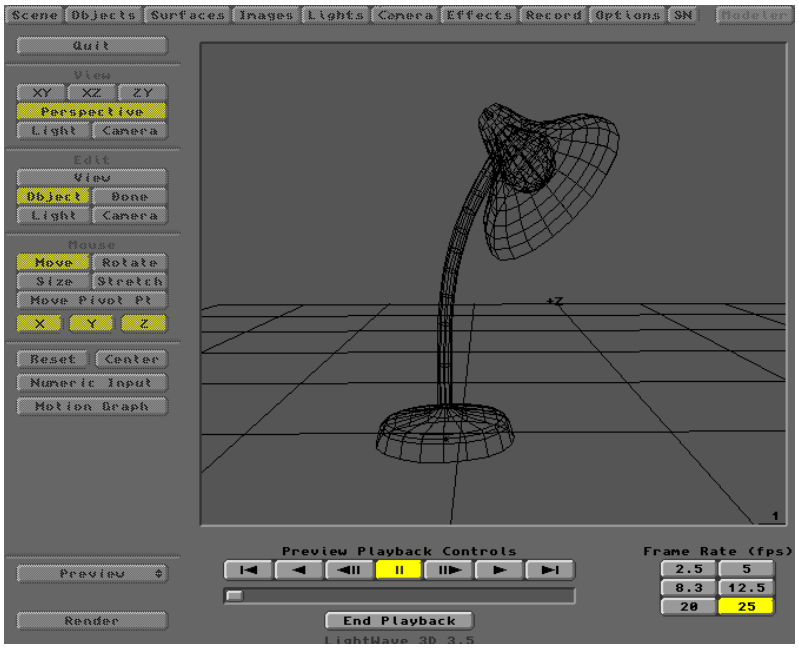
Save Framestores: Indicates *On* or *Off*, according to whether *Framestore* saving has been invoked under the *Record* menu (see below).

Data Overlay: Indicates *On* or *Off*, according to whether *Data Overlay* has been invoked under the *Record* menu (see below).

Serial Recording: Indicates *On* or *Off*, according to whether *Serial Recording* has been invoked under the *Record* menu (see below).

Click *OK* when you are happy with the contents of the *Render Scene* panel and you wish rendering to commence. Otherwise, click the *Cancel* button to cancel the rendering process.
Note: When you click on the *Render* button, the screen will display the progress of the render as a grey-scale image. This display provides information on the *Frame Number*, *Rendering Time*, etc. Once the first segment of the render is complete, the screen will switch to a HAM6 or HAM8 display, according to the selected option. Please refer to the *Segment Memory* notes under the *Camera Menu* for a description of the segment size. HAM images are displayed in segments as they are completed. You can switch between the grey-scale and HAM display using the RMB or using the (Left Amiga/M) key combination. Rendering is aborted by pressing the *Esc* key *with the cursor placed within the display area*.

Preview Playback Controls



The Preview Control panel pops up automatically when LightWave has compiled the *Preview*. It gives you full control of the animation. There are seven control buttons. Three control forward replay, three handle reverse replay and one is the *Pause* button. Their functions are listed below.

Reverse Once — Click to play the animation backwards once.

◄	Reverse Play	Click for repeated backward play.
◄II	Reverse Stop	Click to step backward one frame.
II	Pause	Click to pause current playback.
II►	Forward Step	Click to step forward one frame.
►	Forward Once	Click to play the animation forwards once.
►I	Forward Play	Click for repeated forward play.

Below the *Playback Control* button bank is a *Slide Bar*. This can be used with the LMB to scroll through any number of frames in either direction and to watch playback at any desired speed. This is handy for examining small sections of the animation. At the bottom right of the display is a group of six buttons which control the playback *Frame Rate*.

Frame Rate (fps)

These six buttons provide replay rate options in *Frames per Second* (*fps*). Typically, animations are played at a rate of 25 - 30 fps, corresponding to typical movie film and full motion video. The *Frame Rate* buttons permit *Preview* replay rates from 25, through 20, 12.5, 8.3, 5, to 2.5 frames per second.

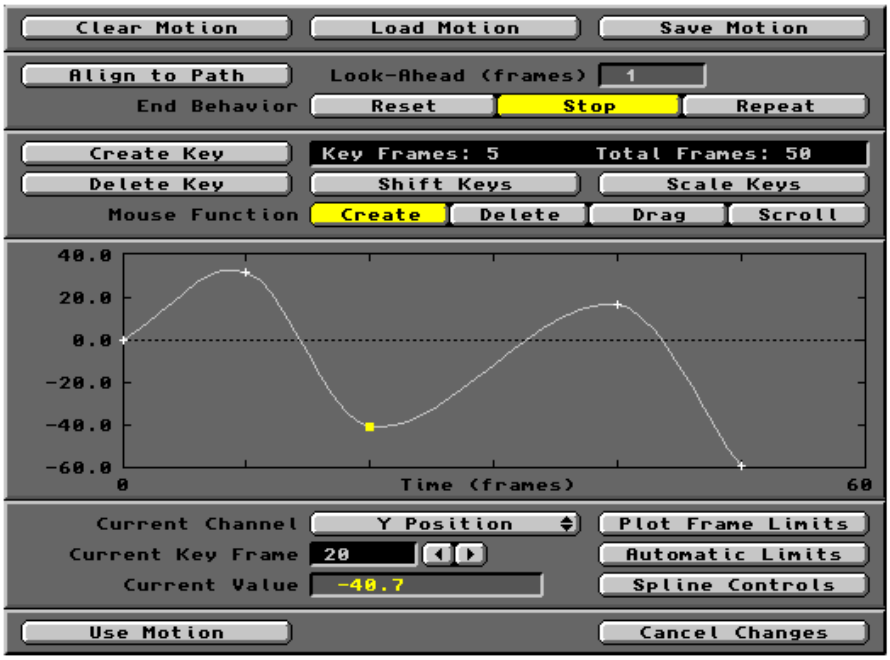
End Playback

Below the *Preview Playback Control* panel is the *End Playback* button. Clicking this ends the *Preview* session and returns control back to the Layout interface. However, the *Preview* animation will remain in RAM and can be viewed again by clicking on *Preview/Play Animation.*

Preview/Free Preview

To remove the animation currently in RAM, select the *Preview/Free Preview* option. This frees up the *Preview* system and permits a fresh *Preview* animation to be generated.

The Motion Control Editor



The format of the *Motion Graph Editor* is shown below and is common to all *Objects*, *Lights* and the *Camera*. The underlying Layout screen has been removed in the interests of clarity.
The *Motion Control Editor* allows you to see the changes in position (*X Position*, *Y Position* and *Z Position*), of orientation (*Heading Angle*, *Pitch Angle* and *Bank Angle*) and of *Velocity*, in a graphical form. The graphs for each channel are displayed individually in the central section of the panel. For *Objects* and *Bones*, additional data are provided on any size (*Scale*) changes (*X Scale*, *Y Scale* and *Z Scale*) undergone by the item. The editor allows you to alter any of the settings displayed on the graph in a similar way to using the *Numeric Input* button on the Layout interface. Additionally, changes may be brought about by editing the *Motion Graph* directly with the mouse. Thus, you can visually fine-tune the item's motion over time, either by changing existing *Key Frame* values or by adding further *Key Frames* directly through the *Editor* panel. A *Motion Graph* (*Path*) can be saved for use with another editable item in the current *Scene.* This allows you to arrange for several items to follow exactly the same *Path*. Alternatively, it can be used separately in a different *Scene*.

The *Motion Graph* consists of a white line, or curve. At the start and end of the curve are white crosses (+), known as *Key Points*, indicating the first and last *Key Frames* in the *Scene*. The currently selected *Key Point* is indicated by a yellow block. Intermediate crosses represent intermediate *Key Points.* The horizontal axis of the graph is always *Time* (*Frames*). The vertical axis is determined by the *Current Channel* selector immediately below the graph. The default channel when the editor first pops up is the *X Position* for the selected item. This is displayed on the scroll bar. This bar will provide a list of all available motion graph parameters by clicking and holding the cursor on it with the LMB. The command buttons found on the *Motion Control Editor* are described below.

Clear Motion

Clears all the *Motion* settings of the currently selected item. The item will relocate to its as-loaded position on the Layout *Grid*. This button only clears the *Motion* held in RAM, it does not delete any data on the hard drive. *Motion* files are stored in the *3D:Motions* directory.

Load Motion

Loads a *Motion* file stored on the hard drive. Clicking on the *Load Motion* button pops up a file requester which defaults to the 3D:Motions directory (also see *LW-config* above). Use this to locate the appropriate *Motion* file. Click on *OK* to load the file. The corresponding *Motion Graphs* are created on the *Motion Graph Editor* panel. Note that a M*otion* file may be *Loaded* into a *Scene* containing a different total number of *Frames* from that which created the file. In cases where the *Scene* has more *Frames* than the *Motion* file, you may wish to adjust (offset) the positions of the *Key Frames* of the *Scene* so that events occur at the correct time. Do this using the *Shift Keys* command (see below).

Save Motion

Click on this button to *Save* a *Motion* file. The pop-up requester defaults to the *3D:Motions* directory. Enter an appropriate filename and click *OK* to save it.

Align to Path

Align to Path causes a selected *Object* to remain orientated to the line of the *Motion Curve* (*Path*). The orientation of the *Object* along the course of the *Motion Path* will be determined by the *Heading*, *Pitch* and *Bank* of the *Path* at each *Key Frame*. The motion created by *Align to Path* can be refined using the *Look-Ahead* parameter.

Look-Ahead (frames)

Look-Ahead allows an *Object* to 'anticipate' turns in the *Path* ahead and displays a smoother transit around them. An *Object* negotiating changes in direction produces more natural motion if it 'senses' the impending change. It 'prepares' itself for the new direction. The *Look-Ahead* parameter determines how far ahead of the event this anticipation is applied. A typical value is 3 *Frames*. A *Look-Ahead* value of 10 or more *Frames* will cause the *Object* to start turning around its axes long before it actually reaches the change in the *Motion Path*.

End Behaviour

End Behaviour is the setting which determines what the selected item will do when it reaches the end of the *Motion Path*. There are three options as described below.

End Behaviour/Reset *C*auses the (*X*, *Y*, *Z*) positional vales, (*H*, *P*, *B*) orientation values and any *Scale* (size) values to reset to their defaults in the *Frame* immediately following the final *Key Frame* of the *Motion Curve*.

End Behaviour/Stop The *Stop* setting will cause the selected item to remain at the location and orientation it had at the last *Key Frame* in the *Motion Curve*.

End Behaviour/Repeat *Repeat* will cause the selected item to repeat its *Motion* immediately it reaches the end of the *Motion Path*. To *Repeat* the behaviour, the *Motion Curve* must contain fewer *Key Frames* than the *Scene.*

Create Key

Click this button with the LMB to *Create* a new *Key Frame*. Enter an appropriate number in the pop-up panel and click *OK* to confirm. The *Motion Graph* will be extended to the new *Key*. Its parameter values will be the same as the previously selected *Key*. Adjust them as required (see *Mouse Function* below). To the right of this button, a small black screen displays the number of *Key Frames* in the current *Motion Graph* and the total number of *Frames* covered by the *Motion Graph*.

Delete Key

Click this button to *Delete* a *Key Frame*. Enter its number in the pop-up panel. The default is the current *Key Frame*. Click *OK* to confirm the deletion. You cannot *Delete* the first *Key Frame* in a *Motion Graph*. Frame 0 is usually a *Key Frame*.

Shift Keys

Clicking this button pops up the *Shift Keys* panel. This allows you to *Shift* (increase or decrease) the *Key Frame* numbers for the current channel. This *Shift* can be applied to all *Key Frames* in the *Motion Graph* or a selection of them. There are four fields into which you may enter new data via the keyboard.

Low Frame This is the number of the first *Key* in the series to be *Shifted*.

High Frame This is the number of the last *Key* in the series to be *Shifted*.

Shift Frames by This is the extent of the *Shift* for each *Key* within the selected range. You may enter a positive integer (increases) or a negative integer (decreases).

Shift Values by This is the change (positive or negative) you wish to apply to the exiting value for each *Key* in the current channel.

After entering the appropriate data, click *OK* to confirm and the changes will be seen in the *Motion Graph*.

Scale Keys

Clicking this button pops up the *Scale Keys* panel. This allows you to *Scale* (stretch or compress) the *Motion Path* for the current channel using a larger or smaller number of *Frames*. The number of *Key Frames* remains the same, they are simply spread apart or brought closer together within a new *Frame* range. The changes are brought about by entering a multiplier (*Scale*) value. This may be an integer, a decimal fraction or both.
A *Scale* change can be applied to all *Key Frames* in the *Motion Graph* or a selection of them. There are four fields into which you can enter new data via the keyboard.

Low Frame This is the number of the first *Key* in the series to be *Scaled*.

High Frame This is the number of the last *Key* in the series to be *Scaled*.

Scale Frames by This is the extent of the *Scale* change for each *Key* within the selected range. You may only enter a positive number.

Scale Values by This is the multiplier you wish to apply to the existing value for each *Key* in the current channel. You may only enter a positive number.

Note: If you apply a fractional *Scaling* factor, some *Key Frames* may be *Scaled* until they are adjacent to one another. You cannot down *Scale* such *Keys* any further.

Mouse Function

The *Mouse Function* buttons change the function of the mouse for editing the *Motion Graph* interactively. Note that your editing of the *Graph* invokes automatic *Spline* adjustment of the curve. As you change the coordinates of *Key Frames*, LightWave will calculate the optimum *Spline Curve*. You can change this later using the *Spline Controls* button (see below).

Mouse Function/ Create This button allows you to *Create* a *Key Frame* using the mouse. Move the cursor to the desired location and click with the LMB. A new *Key* will be *Created* at the numerical value and *Frame* number closest to that location. The new *Key Frame* will be the selected *Key*.

Mouse Function/ Delete This button allows you to *Delete* a *Key Frame* using the mouse. Move the cursor to the desired cross on the *Motion Graph* and click the LMB. The *Key* is selected and immediately removed. The next higher *Key* becomes the selected *Key Frame*.

Mouse Function/Drag Clicking this button allows you to *Drag* a *Key Frame* along either axis of the *Motion Graph*. To alter the parameter *Value* use the LMB. To change the *Frame Number* use the RMB. *Dragging* a *Key* along the *Time (frames)* axis is only possible between the next higher and the next lower *Keys*.

Mouse Function/Scroll Click this button to allow you to *Scroll* the *Time (frames)* axis of the *Motion Graph.* Use the LMB to move the entire *Motion Graph* display left or right. You cannot *Scroll* below the first Frame (0).

Current Cannel

The *Current Channel* button displays the currently selected parameter. Click on the scroll bar with the LMB to display all the parameters available for the selected item. When the cursor is pointing at the desired parameter, release the LMB. The button displays the selected parameter for the *Motion Graph* display.(Also see the introduction to this chapter)

Current Key Frame

This window displays the number of the currently selected *Key Frame*. You can skip between *Keys* by clicking on the left or right skip button. The *Current Key* indicator will skip to the selected *Key Frame* number.

Current Value

This window displays the *Value* assigned to the *Current Key Frame*. You can alter this by typing in the required *Value* using the keyboard. The *Graph* will update interactively.

Plot Frame Limits

Click this button to pop up the *Plot Frame Limits* panel. Here, you can enter (keyboard) the lower and upper *Frame* numbers for the *Time (frames)* axis. The *Motion Graph* display will limited to those *Frames*. This facility allows you to 'zoom' into a small section of a *Motion Graph*, or to display the whole contents of an extensive *Motion Graph*.

Automatic Limits

The *Automatic Limits* button will automatically fit an entire *Motion Graph* within the display area, irrespective of the number of *Key Frames* it contains.

Spline Controls

Clicking the *Splines Controls* button pops up the *Current Key Frame Spline Controls* panel. For details on this editor and on *Splines* in general, see the description of Layout's *Spline Controls* below.

Use Motion

Click the *Use Motion* button when your editing of the *Motion Graph* is complete and you wish to incorporate it into the current *Scene*. This command replaces the data in RAM with the new *Motion* file. The *Motion Graph* editor is closed and you are returned to the Layout screen. The *Scene* in RAM will incorporate the new *Motion Graphs*.

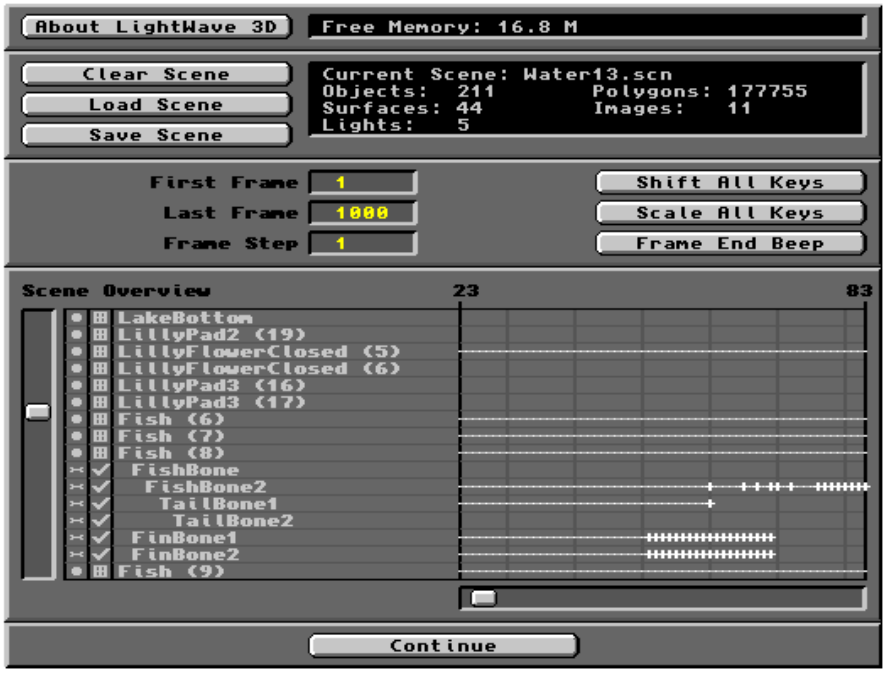
Cancel Changes

Click this button to cancel all changes and return to the Layout display.

Layout Menus

Along the top edge of the Layout screen is a row of ten *Menu* buttons. Clicking on one or other button in this group pops up an editable panel. Here, various parameters connected with the selected menu can be set up. All menu panels pop on top of the ghosted Layout screen. In the graphics reproduced below, the underlying screen has been removed for clarity, though the upper row of menu buttons is retained to show their selected/highlighted state.

The Scene Menu



Clicking on the *Scene* button pops up the *Scene Menu* panel, shown above. From here, you can load a *Scene* prepared and saved earlier, clear the current *Scene* from RAM or save a *Scene* you have edited. You can also edit the *First Frame*, *Last Frame* and *Frame Step* parameters used in the wireframe *Preview* animation and in the full rendering process. Each button and field on the panel is described below.

About LightWave 3D

Click on this button to get some information about the software.

Free Memory:

This small window displays the available free RAM (in Megabytes) before or after loading a *Scene* file. For maximum enjoyment of LightWave, it is suggested that a SIMM of *at least* 16MB is used. Pc users need considerably more.

Clear Scene

Click here to *Clear* any *Scene* data from memory. A requester will pop up asking for confirmation. Click on *Yes* or *No* as appropriate. *Clear Scene* does not delete any files from the hard drive, it simply resets the Layout interface to its default values.

Load Scene

Click *Load Scene* to load a *Scene* file from the hard drive *3D/Scenes* directory. A *File Requeste*r will pop up asking you to select the desired *Scene* file. Select the file with the mouse, or enter the *Path* to the file as necessary. When the required *Scene* file has been selected, click *OK* to Load it, or click *Cancel* to cancel the *Loading* process. You may *Load* a fresh *Scene* without *Clearing* the current *Scene*. The new *Scene* will replace the existing data in RAM. If you wish to replace the current *Scene*, ensure any changes you have made to it are *Saved* (if appropriate) before *Loading* the new *Scene*. The *Scene* file contains a record of all the required *Objects*, though it does not store them directly. *Objects* are recalled from the *3D/Objects* directory via the *Scene* file's script. Similarly, *Surface* data are recalled from the *3D/Surfaces* directory.

Current Scene: This field displays the filename of the current *Scene*.

Objects:
Surfaces:
Lights: These display the total number of each item type used in the current *Scene*.
Polygons:
Images:

Save Scene

After creating or editing the current *Scene*, the data held in RAM may be *Saved* by clicking the *Save Scene* button. A file requester will pop up asking you to select an appropriate filename from the list, or you can type in a new name using the keyboard. Click *OK* to *Save* the file, or click *Cancel* to cancel the *Save Scene* process.

Below the *Current Scene* statistics window are three fields which you may edit for the creation of a new animation. The insertion point for numeric data can be made using the RMB. Data entered here are also used in the wireframe animation *Preview* routine.

First Frame This field defaults to 1, the normal starting *Frame* for LightWave renders. The *Starting Frame* number can be changed by entering a new number using the keyboard and pressing the *Return* key.

Last Frame This is the number of the last *Frame* of the animation to be rendered. Insert the desired *Frame* number using the keyboard and press *Return*.

Frame Step This is the step made by the rendering process in moving from one *Frame* to the next. This field defaults to 1, but can be set to any suitable value via the keyboard.

Shift All Keys

Clicking this button pops up a warning that every *Motion* and *Envelope* in the *Scene* will be affected. Clicking *Yes* will pop up the *Shift All Keys* panel. This allows you to *Shift* the *Key Frame*s forward or backward in time. This may be applied to all *Keys* in the Scene, or a selection of them falling between specified *Key Frame* limits. The result is that the changes covered by the *Keys* will occur either sooner or later in the *Scene*. This is achieved by the insertion or removal of a block of *Frames* before or after the specified range. There are three fields into which you may enter new data via the keyboard.

Low Frame This is the number of the first *Key* in the series to be *Shifted*.

High Frame This is the number of the last *Key* in the series to be *Shifted*.

Shift Frames by This is the extent of the *Shift* for each *Key* within the selected range. You may enter a positive integer (postpones) or a negative integer (advances).

Of many possible time-related events built into a *Scene*, only *Motion Curves* and *Envelopes* are affected by the *Shift All Keys* operation. However, if you postpone an event to beyond a later *Key Frame*, the latter (and any other *Keys*) will be *Shifted* automatically, so that the changes they control occur at a later time. Click *OK* to complete the *Shift*. The *Scene* panel's *Last Frame* field will change to accommodate the *Shifted Keys*.

Scale All Keys

Clicking this button pops up a warning that every *Motion* and *Envelope* in the *Scene* will be affected. Clicking *Yes* will pop up the *Scale All Keys* panel. This allows you to *Scale* the *Key Frame*s and thereby compress or expand the events they control. This may be applied to all *Keys* in the Scene, or a selection of them falling between specified *Key Frame* limits. The result is that the changes covered by the *Keys* will occur either faster or more slowly in the *Scene*. This is achieved by the insertion or removal of *Frames* within the specified range. There are three fields into which you may enter new data via the keyboard.

Low Frame	This is the number of the first *Key* in the series to be *Scaled*.
High Frame	This is the number of the last *Key* in the series to be *Scaled*.
Scale Frames by	This is the extent of the *Scale* change over the selected range. You may only enter a positive number. Numbers greater than 1 will expand the range, causing events to occur more slowly. Numbers less than 1 will compress the range by the factor used, causing events to occur more quickly.

Frame End Beep

Clicking on this button causes LightWave to emit a beep after each *Frame* is rendered, in either *Manual* or *Automatic* mode. The beep is output via the Amiga's audio sockets.

Scene Overview

The lower half of the *Scene Editor* provides an overview or synopsis of all *Objects*, *Bones*, *Lights*, the *Camera* and their activity throughout the *Scene*. The vertical slider can be used to scroll through a *Scene* containing more than sixteen items. The horizontal slider can be used to scroll through a *Scene* containing more than sixty *Frames*.

The main area of the *Editor* panel provides a list of all items in the *Scene* by name. Adjacent to the item list is a *Frame* counter divided into bands according to the total number of *Frames* in the *Scene*. The *Frame* counter is marked with a white cross at each *Key Frame* for each item. Items with a *Motion Path* have their *Keys* joined with a broken line. Any item in the list may be selected directly by clicking either on its name, or its symbol. The selected item is highlighted in yellow.

Next to the vertical slider is a column containing small symbols. These designate the type of the item named in the list. Next to this is a second column with symbols designating the visibility of each item in the Layout window. The visibility status can be adjusted by repeatedly clicking on the symbol using the LMB. The status symbol will cycle through the options. Items not currently visible in Layout have no symbol against them. Switching item visibility when editing highly populated *Scenes* can be helpful to avoid cluttering the Layout window.

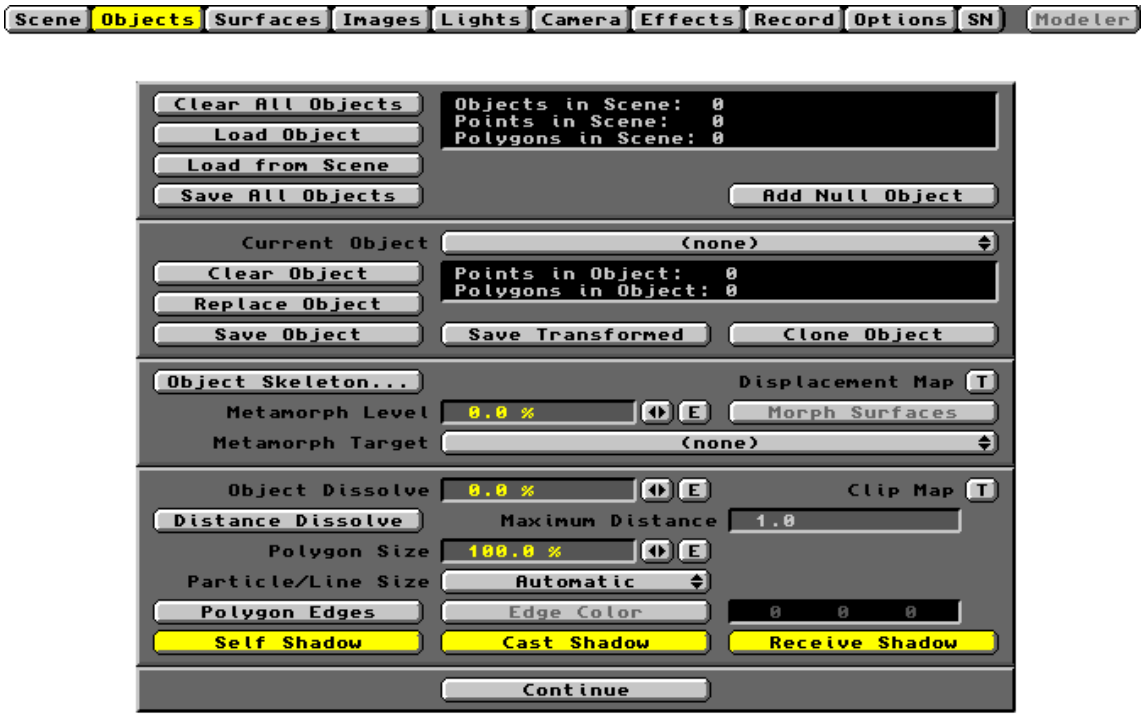
Item Symbol	***Item Visibility***
Objects are designated..... •	*Light* or *Bone* is visible.....
Lights are designated.....	
Only *Bounding Box* visible.....	
The *Camera* is designated.....	Only *Points* visible.....
Bones are designated.....	Some *Polygons* visible (*).....
	All *Polygons* visible.....
	(*) Those facing the X axis

The names of Items that are *Parented* appear indented immediately below their *Parent* item in the *Scene Overview* list.

Continue

Click here to return to Layout.

The Objects Menu



Clicking on the *Objects Menu* button pops up the *Objects Editor* shown above. Using this, you can load *Objects* into the current *Scene*, save *Objects* that you have modified using the editor, replace an *Object* with another, import *Objects* from another *Scene*, remove an *Object* from the current *Scene*, as well as cause one *Object* to metamorphose into another *Object*. You can duplicate (clone) *Objects* and control the way they produce or receive shadows when rendered. The *Objects Editor* also enables you build up an *Object Skeleton* using *Bones*. All buttons have a keyboard equivalent, a list of which can be accessed through the *Help* key. A description of each button on the editor is given below.

Clear All Objects

This button removes all *Objects* from the current *Scene*. A warning pops up asking for confirmation. Click *Yes* to *Clear All Objects*. The *Camera* and *Lights* remain in place, as do any *Images* incorporated into the *Scene*. Only the *Scene* in RAM is affected. The *Scene* file held in the *3D:Scenes* directory is not altered until a *Save Scene* operation is performed.

Load Object

Clicking this button pops up the *Load Object File* requester, which uses the default path to the *Objects* directory. There are sub-directories within the *Objects* drawer, which allow you to keep specific kinds of *Objects* together. Select the required *Object* file and click *OK* to load it into the current *Scene*. The requester will close and you are returned to the *Objects Editor.*

The *Object* file loads into the *Scene* and any associated *Surfaces* or *Images* will be loaded with it. If the associated files are not found in the pathway stored within the *Object* file, a message will pop up asking if you wish to select an alternative. Click *Yes* or *No* as appropriate. As the *Object* file loads, the information window at the top of the panel will show progress. The window contains information on the total number of *Objects*, total number of *Points* and total number of *Polygons* in the current *Scene*.

The freshly loaded *Object* becomes the currently selected *Object* and its name will appear in the *Current Object* scroll bar. The total number of *Points* and *Polygons* contained in this *Object* are given it the second information window. LightWave can import *Objects* constructed via a range of software packages. If it is uncertain about a foreign file format, a menu will be displayed from which the appropriate filetype may be selected. Files which are not compatible will throw up an error message. File format conversion is handled by converter modules kept in the *TIO* directory. Many 3D file formats can be converted into LightWave *Objects* using a range of converters available commercially and in the Public Domain.

Load Object From Scene

This button allows you to import all the *Objects* from a *Scene* file on the hard drive into the current *Scene*. The imported *Objects* are complete with their associated *Surface* names, *Motion* paths and *Image*, etc. data. *Objects* cannot be imported on an individual basis. Clicking the button pops up the *Load Objects from Scene File* requester, which defaults to the standard location for *Scenes*, the *3D:Scenes* directory. Select the appropriate *Scene* file and click *OK* to proceed. A query panel will pop up asking if you wish to include *Lights* with the importation. Click *Yes* or *No* as appropriate and the requested files will be loaded into the current *Scene*. The *Scene Editor* panel will update accordingly.

Save All Objects

Click this button to *Save All Objects* in the current *Scene*. Their current *Surface* names will be saved with them. A query panel will ask you to confirm the action, after which all *Objects* will be saved to the hard drive in their original locations. Duplicate *Objects* (clones) should first be given separate names if you wish to save each one, otherwise only a single file will be kept. This principle also applies to their *Surface* names.

Add Null Object

The nature of *Null Objects* is given in the *Preamble*. Please read that section to gain an understanding of their properties and purposes in *Scenes*. Click on this button to load a *Null* into the current *Scene*. The *Scene Editor* confirms the addition by placing the name *NullObject* in the *Current Object* scroll bar.

Current Object

The *Current Object* scroll bar contains the name of the currently selected *Object*. To change the *Current Object*, place the cursor on the bar and press the LMB. A pop-up list of all *Objects* in the *Scene* will appear under the cursor. While holding down the LMB, move the cursor to the required *Object* and release the LMB. The required *Object* is now the *Current Object*. Only the *Current Object* is affected by the buttons in this and lower sections of the *Editor* panel.

Clear Object

Click this button to *Clear* the *Current Object* from the *Scene*. Only the *Object* held in RAM is removed. The *Object* file held in the *3D:Objects* directory remains unaffected.

Replace Object

Click this button if you wish to replace the *Current Object* with another from the *3D:Objects* directory, or other source. The *Replace Objects File* requester pops up, into which you should enter the replacement *Object's* name. Click *OK* to confirm. The requester will close and the *Object Editor* will update to the replacement *Object* as the *Current Object*. This tool is handy for conducting trial renders or making *Preview* animations which contain a highly complex *Object*. Complex *Objects* take much longer to process than simple *Objects*, so *Replace* it with a simpler *Object* until you are happy with the *Scene*, then *Replace* the temporary *Object* with the original.

Save Object

Save Object saves the *Current Object* with its *Surface* settings only. It does not save any changes in the *Size* or *Stretch* parameter that you may have applied after *Loading* it, or the *Scene*. *Objects* are *Saved* with their default (as *Loaded*) dimensions intact. Use the *Save Object* button to save any new *Surface* textures you have applied

using the *Surfaces Editor* (see below). The *Save Scene* button (see above) does not save *Surface* data, these are saved within the *Object* file. *Save* cloned *Objects* under different names if you wish to retain their individuality.

Save Transformed

The physical shape of an *Object* can be modified within Layout using *Displacement Mapping* or by the use of *Bones*. Any *Object* thus modified from its original shape, can be saved using the *Save Transformed* button. A *Warning* will pop up suggesting that you use a different name for the *Transformed Object* if you wish to retain the original. Closing this pops up the *Save Transformed Object File* requester. This also defaults to the *3D/Objects* directory, to which you can *Save* the *Transformed Object* by entering an appropriate filename and clicking *OK*. Note that this action saves the *Object's* location and orientation relative to Layout's *Origin*, just as *Objects* are saved by Modeler.

Clone Object

To make identical copies of the *Current Object*, click the *Clone Object* button. A panel pops up asking for the *Number of Clones* to make. The default is 1. Enter the required number and click on *OK*. You will be returned to the Object Editor, which will update the *Current Object* lister to show a series of clones numbered (1), (2), etc. The cloning process copies all the *Surface*, *Motion*, *Morphing*, etc. settings of the original *Object*.

Object Skeleton

Click this button to gain access to LightWave's *Bones* function. The principle of using of *Bones* is discussed in the *Preamble*. Please read it to ensure you understand following information.

Clicking *Object Skeleton* pops up the *Skeleton for "Object"* control panel. Use this to add, remove, name, activate, position, orientate and define the scale of influence of *Bones*. A detailed description of this panel is given below.

Metamorph Level

Metamorphosis is discussed in the *Preamble.* Please read through that account so that the following information enables you to use LightWave's metamorphing ability with confidence. For convenience, the word *Metamorph* is often shortened to '*Morph*'. The *Metamorph Level* indicates the percentage to which the *Current Object* is metamorphosed (morphed) into the *Metamorph Target Objec*t when the *Frame* is rendered. The resulting *Object* will be termed the *Morph Object* hereafter. You can insert any value between 0.0% (no metamorph) and 100% (complete metamorph). A *Metamorph Level* of 100% is equivalent to using the *Target Object* instead of the *Current Object*. The numeric data can be inserted directly using the keyboard. Alternatively, click on the adjuster button (**<>**)and drag the mouse right (to increase value) or to the left (to decrease value).

Clicking the *E* button pops up the *Envelope Editor* panel (see below). This enables you to conduct the *Morph* over a specified number of *Frames* and to follow a specified course or *Envelope* . The *Envelope* provides for an animated *Morph*, which could if need be, return to the original *Object*. Any metamorphosis from the *Current Object* into the *Target Object* has an associated *Morph Envelope*, whether or not you create one using the *Envelope Editor.* The *Envelope* produced by setting a *Metamorph Level* of, say 50%, is a horizontal line at 50% throughout the animation and the same *Morph Object* is seen throughout. *Envelopes* always over-ride the value set in the numeric field.

Metamorph Target

The *Metamorph Target* is the *Object* into which the *Current Object* will be morphed according to the *Morph Envelope*. The *Metamorph Target* may be any *Object* which fulfils the morphing criteria (see *Preamble* on Metamorphosis). There can be up to forty (sequential) *Metamorph Targets* for any single starting *Object* and all *Objects* in the *Scene* may be morphed. Their *Morph Envelopes* can be set up to occur concurrently (parallel morphs) or sequentially (serial morphs) or a mixture of both. All *Metamorph Targets* must be loaded into the *Scene's Object* listing though they need take no formal part in the action. Indeed, it is often appropriate to place *Targets* out of camera view, or make them 100% dissolved (i.e. invisible), throughout the animation. *Targets* simply provide their structural data which LightWave uses to compute the *Morph*.

Morph Surfaces

The *Morph Surfaces* button is a toggle, which when active arranges for the *Surface* settings of the starting *Object* to be morphed into those of the *Morph Target* . The *Surfaces* morph will be controlled by the same *Morph Envelope*. If the *Morph Surfaces* button is not highlighted, then the *Morph Object* will retain the *Surface* characteristics of the starting *Object*.

Displacement Map

The *Displacement Map* button is labelled with a *T*, which indicates that a *Texture* mapping routine will be invoked. Whenever you click on a *T* button, the *Texture Mapping Editor* panel pops up (see below). *Displacement Mapping* is the process of displacing the elevation of *Polygons* according to the greyscale values

contained in an *Image* or in a mathematically generated *Procedural Texture.* When the *Displacing Mapping* routine is complete, the *Object* 's surface will be deformed into another shape. This new *Object* can be *Saved* using the *Save Transformed* button described earlier (see *Objects Editor* above). *Displacement Mapping* is included within the skeletal deformation and morphing group because its effects are analogous. A new *Object* shape results and the change can be animated.

Object Dissolve

Object Dissolve is a numeric data field whose value determines the visibility of the *Current Object* when the *Scene* is rendered. A completely *Dissolved Object* is invisible when rendered. The *Dissolve* parameter in this case would be 100%. Conversely, a *Dissolve* parameter of 0.0% means that the *Object* is completely visible when rendered. Set the required value either by inserting it with the keyboard, or by clicking and holding down the LMB on the **<>** button. Drag to the left (decreases the value) or the right (increases the value). When the required value is showing, release the LMB. Any *Object Dissolve* can be animated using an *Envelope*. Click on the *E* button to access the *Envelope Editor* panel (see below).
The *Object Dissolve* tool allows you to create ghost-like images, or semi-transparent surfaces. Using the *Dissolve* function is somewhat analogous to applying a *Transparency* value to the *Object's* surfaces via the *Surfaces Menu*. However, a multi-surfaced *Object* would require each and every different *Surface* to be given a *Transparency* value or *Envelope*. Using *Object Dissolve* enables you to achieve the same result by using just one setting or *Envelope*. The *Envelope* profile over-rides any value set in the value field.
Note: Any *Object* given a 100% *Dissolve* value will not be drawn in the Layout window and its Bounding Box will appear in dashed lines.

Distance Dissolve

Distance Dissolve is a special form of dissolve in which an *Object's* visibility is controlled by its distance from the *Camera.* The *Maximum Distance* field controls how far away from the *Camera* it must move to become totally dissolved (invisible). Any position between that distance and the *Camera* will show a transparent image. The image density varies with distance and is a useful aid for rendering underwater *Scenes*.

Maximum Distance

This is a numeric data field which defaults to 1.0. This is analogous to 1.0 unit of the *Grid* setting.

Clip Map

A *Clip Map* is an *Image* or procedural *Texture* map, which is used to 'remove' portions of the *Current Object* according to the grey-scale values it contains. *Clip Mapping* is particularly useful when you want to generate lifelike shadows of *Objects* which contain partially or completely transparent elements. Transparent elements should not cast a shadow. However, when Layout renders a 'transparent' *Object*, it still 'sees' the *Polygons*, irrespective of any *Transparency* or *Dissolve* values. This causes the *Shadow Mapping* routine to generate an unwanted shadow. A black *Clip Map* which corresponds to the transparent portions of the *Surface* will 'remove' the relevant *Polygons* from the 'view' of the *Light* source. The result is a perfect shadow. Click the *T* button to access the *Texture Mapping* panel (see below) via which an appropriate *Image* can be utilised.

Polygon Size

The *Polygon Size* field indicates the percentage *Size* of the *Polygons* in the *Current Object* relative to their normal 'full size'. Thus a value of 100% means the rendered image will appear normal. Values less than 100% cause the rendered *Polygons* to shrink, breaking up the integrity of the *Object*. The value is inserted using the keyboard, or by clicking and dragging (LMB) the adjuster button (**<>**), or using an *Envelope* to animate the change. The rendered *Polygons* retain their original shapes and locations in 3D space, but they get smaller by contracting towards their centres. Animation of *Polygon Size* using an *Envelope* (*E* button), allows for an *Object* to explode or disintegrate into dust.

Polygon Edges

The edges of *Polygons* (the mesh) are not rendered in normal circumstances. Rendered images are usually obtained from the *Surfaces* of *Polygons*. However, the *Polygon Edges* button toggles edge rendering on or off. The default is off. Click on the button to highlight it (on) and the previously ghosted *Edge Color* button is enabled, together with its *RGB* colour indicator. Click the *Edge Color* button and the *Polygon Edge Color* control panel pops up. Here, you may use the sliders to adjust the *Red*, *Green* and *Blue* components of the required *Edge Color*. When you are happy with this, click *OK* to return to the *Objects Editor*. When you render an image so prepared, the mesh of the *Current Object* will be drawn in the selected *Edge Color*. The *Polygon Edges* toggle is used in conjunction with the *Particle/Line Size* button.

Particle/Line Size

The *Particle/Line Size* scroll bar is only active when *Polygon Edges* is enabled. This button allows you to adjust the size of any *Single Point Polygons* and the thickness of the lines used in the *Polygon Edges* render. There

are four options available when you click on the bar with the LMB. *Small*, *Medium* and *Large* indicate the nominal thickness of the rendered mesh. The *Large* and *Medium* options can cause a cluttered image to be rendered, especially in areas of high *Polygon* density. Click on *Small*, or to be sure, use the *Automatic* (default) option, which scales the thickness for optimal clarity.

Edge Color

The *Edge Color* button is enabled when you activate *Polygon Edges*. Adjust the *Edge Color* as described above.

The last three buttons on the *Objects Editor* control the ability of the *Current Object* to influence the rendering of shadows. These should be used in conjunction with the shadow casting routines to be found in the *Lights Editor*. To ensure you get the results you expect, remember that shadow generation is a function of both the *Light* source and the *Object* onto which the light falls.

Self Shadow

Click (highlight) this button to enable the *Current Object* to cast a shadow on itself. The default is on, though not all *Objects* are able to do this. A sphere, for example, will not be affected.

Cast Shadow

Click this button to enable the *Current Object* to cast shadows onto other *Objects* in the *Scene*.

Receive Shadow

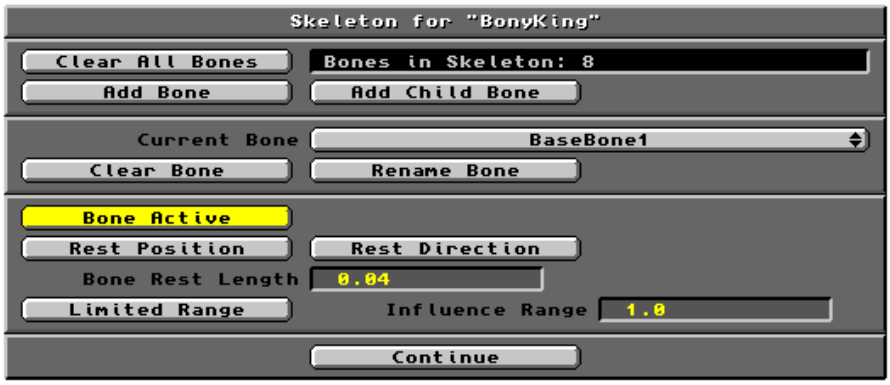
To allow the *Current Object* to *Receive Shadows* from other *Objects*, this button must be active.

Note that *Casting* and *Receiving Shadows* are complementary functions which you must satisfy for all the *Objects* in which shadowing is involved. Also remember that *Shadow Mapping* is a very time consuming exercise. Only activate shadows on the most important elements of the *Scene*.

Continue

Click here to return to the Layout interface.

Bones and the Skeleton Control Panel



This panel pops up when you click the *Object Skeleton* button on the *Objects Editor*. It is used for adding and defining the properties of *Bones.* These are manipulated into an *Object Skeleton* using the Layout controls.

Following is a description of all the buttons and fields on the *Skeleton* control panel. Tutorial 5 provides a practical and detailed insight to the implementation of *Bones* in an animation project.

Clear All Bones

Click this button to remove all the *Bones* assigned to the *Current Object*. Note that *Bones* are not saved as discreet items, but form part of the *Scene* file. Also note that the *Clear all Bones* command is irreversible. Therefore, any mistake using *Clear All Bones* will require a complete re-*Load* of the *Scene* or a re-work of the *Skeleton*. The *Scene* file held in the *3D:Scenes* directory, which contains all *Bones* data for the *Object's Skeleton*, remains unaffected. Only if the *Current Scene* is saved will the *Bones* data be updated to the hard drive. A query panel will ask for confirmation before you click *Yes* to complete the action.

Add Bone

Click this button to *Add* a *Bone* to the *Current Objec*t. On doing so, the *Current Bone* scroll bar will be filled with the default name *'Bone'*. To help you with its use, each added *Bone* should be given a meaningful name using the *Rename Bone* button. All freshly added *Bones* are placed at the *Origin* of the Layout *Grid.* Relocate a new *Bone* to the desired position in the *Object* using Layout's *Edit/Bone* controls.

Add Child Bone

Add *Child Bone* will add a *Child Bone* to the *Current Bone*. The *Current Bone* name will change to *'Bone'*, the default name for all new *Bones*. Change this name as appropriate using the *Rename Bone* button. A *Child Bone* is added to the sharp end of the *Parent Bone*. In this way, chains of child *Bones* can be built up to form a backbone-like structure. The *Parent* of any *Child Bone* is the one it is connected to at its blunt end.

Bones Information Window

Located just above the *Add Child Bone* button, this displays the total number of *Bones* in the skeleton of the *Current Object.*

Current Bone

The *Current Bone* scroll bar contains the name of the currently selected *Bone*. To change the *Current Bone*, place the cursor on the bar and press the LMB. A pop-up list of all *Bones* within the *Current Object* will appear

under the cursor. While holding down the LMB, move the cursor to the required *Bone* and release the LMB. The required *Bone* is now the *Current Bone.* Only the *Current Bone* is affected by the buttons in this and the lower section of the *Skeleton* panel.

Clear Bone

Click this button to *Clear* the *Current Bone* from the *Skeleton*. Only the *Bone* held in RAM is removed, though there is no undo for this command. Mistakes using *Clear Bone* will require a complete re-*Load* of the *Scene* or a re-work of the *Skeleton*. The *Scene* file held in the *3D:Scenes* directory, which contains all *Bones* data for the *Object's Skeleton*, remains unaffected. Only if the *Current Scene* is saved will the *Bones* data be updated to the hard drive.

Rename Bone

Click this button to pop up the *Bone Name* edit panel. Enter an appropriate *Name* via the keyboard and press *OK* to complete the renaming process.

Bone Active

Freshly added *Bones* are in an inactive state to enable you to position and size them within the *Skeleton*. Such *Bones* are drawn in dotted lines when selected in the Layout window. When you wish a *Bone* to have influence on the shape of the *Object*, click *Bone Active* to highlight it (yellow). The *Bone* is activated and is drawn in full lines in the Layout window. Active *Bones* cause a deformed *Object* to be drawn in Layout.

Rest Position

The *Rest Position* button pops up a *Bone Rest Position* edit panel. This shows the coordinates of the *Current Bone* at *Rest* and if appropriate, can be edited to an alternative location.

Rest Direction

The *Rest Direction* button pops up a *Bone Rest Direction (degrees)* edit panel. This shows the orientation of the *Current Bone* at *Rest* and if appropriate, can be edited to an different orientation.

Bone Rest Length

This data field indicates the *Rest Length* of the *Current Bone*. This is the length of the *Bone* after any resizing brought about by the *Rest Length* controls in the Layout/*Mouse* group.

Limited Range

The *Limited Range* button toggles LightWave's the ability to limit the extent of a *Bone's* influence on the *Object*. A freshly added *Bone* has an unlimited influence on the *Object* to which it is assigned. However, the usual requirement is for the *Bone* to affect only the part immediately around it and let adjacent or *Child Bones* take care of adjacent areas. This leads to the most life-like *Object* deformation. To enable you to impart this behaviour, you must limit the *Bone's* influence by turning on (highlighting) this button. An alternative strategy is to incorporate several well-oriented *Bones* within the area and allow them to play against each other.

Influence Range

Further control on the extent of a *Bone's* influence can be obtained by setting the *Influence Range* field to a smaller value using the keyboard.

Continue

When you are happy with the *Bone* arrangement in the *Object's Skeleton*, click *Continue* to return to the *Object Editor*.

The Surfaces Menu

The *Surfaces* button gives you simultaneous access to *Object Surfaces* and to *Polygon Surfaces*. This can be slightly confusing, because Layout makes no clear differentiation in its dual use of the word. Remember that *Object Surfaces* are a specific group of *Polygons* which have been given a particular name in Modeler. The *Surface attributes* which you apply to the *Polygons* via the *Surfaces Editor* affect the way they appear when rendered.

The *Surfaces Editor* pops up when you click on the *Surfaces* button. With this, you control all aspects of an *Object's* appearance, other than its shape. Through the *Surfaces Editor* you are able to invest *Objects* with material properties. A *Surface* may be of any colour, it may be reflective, or luminous. They can be transparent and refract light. It can be given textural refinements to make it resemble a natural substance. You may take an *Image* and 'print' it onto any *Surface*, or you can deform a *Surface* visually by bump mapping. The *Surfaces Editor* holds the key to successful image rendering and within it are all the tools you need to bring your *Objects* to life.

Surface attributes can be generated 'on the fly' by manipulation of *Surfaces Editor* tools, or they may be taken from a prepared list. Any *Surface attribute* created in the *Editor* can be *Saved*. The *3D:Surfaces* directory is where LightWave keeps its store of *Surface attributes*. These are data files which can be loaded into the *Editor* and applied directly to any named *Object Surface*. Thus, while the *Object Surfaces* are given logical names (head, arm, leg, etc.) in the Modeler environment, *Surface attributes* (wood, silver, gold. etc.) are defined and saved through Layout. However, a *Surface attribute* and an *Object Surface* may be given identical names to permit logical application and immediate identification. This is especially relevant to *Surfaces* with special or customised attributes. When *Surface attributes* have been assigned to any *Object*, it can be *Saved* as a hybrid file, containing all the surfacing data the *Object* needs. So, whenever that *Object* is *Loaded* into a *Scene*, its *Surface attributes* will be loaded with it.

Each button and field on the *Surfaces Editor* is described below.

Current Surface

The *Current Surface* scroll bar bears the name of the *Object Surface* for which the *Editor* is displaying data. Click on the bar with the LMB and a list of all *Object Surfaces* in the current *Scene* pops up. The list may be extensive and continue beyond the visible panel. If so, an *up* or *down* arrow will indicate that further names are available. The *Editor* will update the displayed data when you change the *Current Surface* name.
Note: The name *Default* is given to any group of *Polygons* which have not been assigned a *Surface* name in Modeler. *Default* always appears in the list, irrespective of whether there are un-named *Surfaces* or not.

Polygon Usage:

The *Polygon Usage* field indicates the total number of *Polygons* in the *Current Scene* that have been given the *Current Surface* name in Modeler.

Load Surface

Any previously saved *Surface attributes* file can be *Loaded* into the *Editor* by clicking on this button. The *Load Surface File* requester pops up and will search the default path for the file list. This is normally kept in the *3D:Surfaces* directory. Select the required file and click *OK* to *Load* the *Surface attributes* file. The *Editor* settings update accordingly, but you will see no reference to the name of the attribute (wood, silver, gold, etc.) . The *Current Surface* name (head, arm, leg, etc.) remains unchanged, because this is the *Object Surface* to which the new attributes will be applied. Note that any *Surface attributes* file can be *Loaded* into the *Editor* and used on the *Current Surface*. The *Current Surface* name will remain in place, while the attributes change to the new values. In this way, several different *Object Surfaces* can be quickly assigned with the same *Surface attributes.*

Save Surface

After editing any data displayed in the *Editor* panel, the *Current Surface* data can be saved to the hard drive. Click the *Save Surface* button and the *Save Surface File* requester pops up. The requester will search the default path for a file list. This is normally kept in the *3D:Surfaces* directory. Either select an existing file to update, or enter a new name using the keyboard. Click on *OK* to complete the process. Please read *Load Surface* above to be clear about *'Surfaces'.*

Rename Surface

The name of the *Current Surface* can be changed by clicking on the *Rename Surface* button. This pops up a *Surface Name* box into which you can type the new name. You may prefer to *Rename* a *Surface* to reflect more clearly the attributes you have assigned to it. For example, a *Surface* named 'hat' might be usefully *Renamed* 'red hat', etc. In this way, several otherwise identical *Objects* may be rendered in different colours, etc.

Alphabetize List

This button toggles the *Current Object* listing in alphabetic or original order. The original order always appears with *Default* first, followed by *Surface* names in the order that they were assigned in Modeler.

Surface Color

Click the *Surface Color* button when you want to flood the *Polygons* of the *Current Surface* with a block of colour. The *Surface Color* control panel pops up, into which you can type or adjust the *RGB* values of the required colour. This is indicated in the colour swatch at the bottom of the display. Use the LMB to drag the *Red*, *Green* and *Blue* component sliders until you are happy with the colour. Click *OK* to return to the *Surfaces Editor*. The numeric values alongside the *Surface Color* button indicate the current colour.

Texture T

Clicking the *Surface Color T* button pops up the Texture *Mapping control* panel (see below). This is used to apply colour to the *Current Surface* using either an *Image* map or a *Procedural* map.

Luminosity

Luminocity enables a *Surface* to be seen in total darkness. Normally, any LightWave *Scene* requires that the Layout stage is lit by strategically placed *Lights*, plus the natural, all-pervasive *Ambient Light*. However If you wish, these *Lights* can effectively be turned off, leaving the stage in total darkness. *Luminocity* allows you to make any *Surface* self-illuminating, so it appears to glow or be lit 'from within'. Enter a value into the field using

the keyboard or by dragging the control button (**<>**) with the LMB. Dragging to the left decreases and dragging to the right increases the *Luminocity* value. Values greater than 100% can only be inserted via the keyboard.

Luminocity T

Luminocity of a *Surface* can be given texture by clicking the *Luminocity T* button. This pops up the *Texture Mapping* control panel discussed in detail below. Only *Procedural Texture*s can be applied. *Luminocity* mapping uses the greyscale values of a *Procedural Texture* to vary the percentage *Luminocity* across the *Surface*.

Diffuse Level

The *Diffuse Level* is the percentage of incident light which is scattered off the *Surface*. In the real world, we are able to see objects in their entirety because of their light scattering properties. Scattered light is diffused in all directions, so you do not have to be positioned at any particular location to see the object. The *Diffuse Level* control enables you to set a value for the percentage of incident light which is scattered and diffused by the *Surface*. Use the keyboard, or the control button (**<>**) to set the value. The default value is 100%, which may be too high for realistic renders. With *Diffuse* set at zero, a non-luminous *Surface* will appear to be black even though you have assigned it a *Colour* value or a *Texture* map.

Diffuse Level T

Click this button to pop up the *Texture Mapping* control panel. Either an *Image* or a *Procedural Texture* can be applied. The *Texture* mapping works by adding luminocity to the *Surface* in accordance with the greyscale (luminance) values of an or a *Procedural Texture*. The *Texture Mapping* control panel is discussed later.

Outline Only

Outline Only renders the outlines (i.e. edges) of *Polygons* rather than their *Surfaces*. The outlines are rendered in the *Surface Color* and any *Texture Mapping* will be seen along them.

Additive

Additive causes the *Colour* of a *Surface* to be added to the *Colour* of any other *Surface* standing behind it in the line of sight of the *Camera*. This is best used with semi-transparent *Surfaces* to give a realistic coloration due to line of sight mixing. Additive causes the apparent brightness and colour saturation to increase and tend towards pure white as more colour components are added.

Sharp Terminator

Sharp Terminator is applied to accentuate shadow along the darker edges of an *Object*. It is particularly useful when rendering 'planets'.

Specular Level

Specular light is reflected from a *Surface* as a cone of rays and appears as a highlight when the *Camera*, *Object* and *Light* source are correctly positioned to each other. Enter the percentage of incident light which will be reflected as *Specular* light using either the keyboard or dragging the LMB on the slider button (**<>**). The character of the *Surface* alters the nature of the highlight. Very hard, shiny *Surfaces* produce small, intense highlights, whereas matt or dull metallic *Surfaces* have larger highlights. You can control this variation with the *Glossiness* control (below).

Specular Level T

The areas of specularity rendered on a *Surface* may be given a texture by clicking on the *Specular Level T* button. This pops up the *Texture Mapping* control panel discussed below. The *Texture* map may be an *Image* or a *Procedural Texture*. The *Texture* mapping works by adding luminocity to the *Surface* in accordance with the greyscale (luminance) values of an or a *Procedural Texture*. The *Texture Mapping* control panel is discussed later.

Color Highlights

The area of specularity on a rendered image is normally given the colour of the *Light* which causes it. You can cause the highlighted area to blend the *Light* colour with the *Surface* colour to create a more realistic highlight. This is particularly used for specular reflections from metallic surfaces. Click the *Colour Highlights* button to produce a colour-blended highlight. If the *Surface* is not allocated any reflective value, an error message will pop up.

Glossiness

Glossiness controls the tendency of a reflective *Surface* to show small or large 'hot spots' of specular light. There are four notional settings for the *Glossiness* parameter, each giving a different specular spread. A *Low Glossiness* produces large areas of specularity. *Maximum Glossiness* gives very small, piercing cones of specularity. Select Low, Medium, High or Maximum according to the desired effect. If the *Surface* is not allocated any reflective value, an error message will pop up when you click a Glossiness button.

Reflectivity

Reflectivity is the ability of a *Surface* to return incident light to its source or emit it at an angle equal to the angle of the incident light. The visual effects of *Reflectivity* are modified by other factors, controlled through the *Specularity Level* and *Glossiness* buttons. The percentage of light reflected by a *Surface* is determined by the *Reflectivity* value. This is entered into the *Editor* using the keyboard, or by dragging with the LMB on the control button (**<>**). Highly reflecting *Surfaces* like chrome and polished silver reflect almost 100% of the incident light. They are often coloured by their surroundings and may reflect distorted images of nearby *Objects*. To do that, you should ensure that the *Trace Reflections* button is activated in the *Camera Editor* (see below). Remember that as the reflectivity approaches 100%, the less can the natural colour of the *Surface* be seen. To do so, use a setting in the *Diffuse Level* field, the *Specular Level* field, or activate *Color Highlights*. A *Surface* with a good *Reflectivity* can be made to reflect an *Image* (see *Reflection Image*), or only the colours of the sky and ground. The latter colours are defined in the *Effects Editor*.

Reflectivity T

Clicking the *Reflectivity T* button pops up the *Texture Mapping* control panel, which can be used to create a variation in *Reflectivity* across the reflection area through the use of an *Image* or a *Procedural Texture* map. The *Texture* mapping works by adding luminocity to the *Surface* in accordance with the greyscale (luminance) values of an *Image* or a *Procedural Texture*. The *Texture Mapping* control panel is discussed later.

Image Seam Angle

Use *Image Seam Angle* button to control the position of the seam in a *Reflection Image* map. The LightWave universe can be visualised as a large sphere, with the Layout stage near the centre. Any *Image* used as a *Reflection Image* is applied like a large, single sheet of wallpaper to the inside surface of the sphere. Thus, all reflective *Objects* inside the sphere will be able to 'see' the *Image*, no matter where they are located or whatever their orientation. At some place on the sphere, the *Image* has a seam. This is where left and right edges are pulled together. Unless the *Image* has matching edges, the seam may be seen as a line in a reflected image. To give you some control in preventing a seam reflection, you can control its location on the sphere by adjusting *Image Seam Angle*. The seam is a vertical line, the position angle of which is analogous to the *Heading*. (If necessary, please refer to *Heading*, *Pitch* and *Bank* in the Layout overview). Thus, a *Seam Angle* of 0º places the seam at the end of the +Z axis, probably its least visible position for reflections. At 90º it is placed to the right end of the +X axis. At 180º the seam is 'behind' (-Z) the default *Camera* position, its most visible position for reflections. At 270º it is to the left, at the end of the -X axis. In most cases, the default *Seam Angle* of 0º will be appropriate. There may, however, be instances where the default position is not ideal.

Reflection Image

Any *Image* loaded into the *Images Editor* (see below) can be used as a *Reflection Image* map. The required *Image* is selected from the list using the LMB to click on the scroll bar. The default *Image* is 'none', which means no *Image* will currently be seen in reflections. Release the LMB when the cursor is on the required name. The scroll bar will update to show the selected *Image* name. The Image will now be seen in reflective *Surfaces* according to the parameters you have set. Furthermore, wherever this *Image* appears in the rendered *Scene*, it will be reversed horizontally to mimic a true life reflection.

Transparency

Transparency controls the percentage of incident light which passes through a *Surface* rather than being reflected or scattered. A setting of 100% renders the *Surface* invisible. Most solid *Surfaces* are opaque and therefore 0% *Transparent.* However, window glass and water, for example, are very transparent and may have *Transparency* values of 90% or more. Light passing through a transparent *Surface* may impinge on another *Surface* located behind it and within the *Camera's* line of site. In such cases, the image of the rear *Surface* may appear *Refracted*, i.e. shifted from its physical position. You can arrange for this to occur by applying a *Refractive Index* to the transparent *Surface*. Set the *Transparency* value by typing an appropriate figure into the field with the keyboard, or click on the adjuster button (**<>**) with the LMB and drag left or right until the desired value is reached and release the LMB.

Transparency T

Clicking on the *Transparency T* button pops up the *Texture Mapping* control panel. Using this you can vary the *Transparency* value over the *Surface* through the application of a *Transparency Texture* map. *Texture* mapping

works by adding *Transparency* to the *Surface* in accordance with the greyscale (luminance) values of an *Image* or a *Procedural Texture*. The *Texture Mapping* control panel is discussed later.

Color Filter

Applying *Colour Filter* allows any *Surface* located behind the *Transparent Surface* to have its colour values *Filtered* by those of the front *Surface*. The result is a realistic, line of site rendering of the rear *Surface* colour as seen through the other. The *Transparency* value must be greater than zero for colour filtering to be active.

Refractive Index

On passing from air into another transparent medium, light rays slow down and are 'bent' out of line due to *Refraction*. The extent of this bending is determined by the *Refractive Index* (RI) of the medium. By definition, a vacuum has an RI of 1.0000. For most practical purposes, air has the same value. Any medium whose RI differs from this will cause light rays to bend as they pass through the interface between the two. Thus, any *Object* seen through that medium will appear distorted and visually repositioned. You see the effect whenever you look into water. In order to render the effects of refraction, you should enable *Trace Refraction* in the *Camera* menu (see below), which must also be set to *Realistic* mode. The RI of many materials is known with accuracy and any you wish to simulate can have their RIs used in Layout's light calculations. A list of the Refractive Indices of several common materials appears below. To be rendered with the *Trace Refraction* function, the RI of a *Surface* must be greater than 0.

Refractive Indices of Various Materials

Vacuum (The standard reference medium)	1.0000
Air	1.0003
Ice	1.310
Alcohol	1.329
Water	1.333
Quartz (varies with type)	1.46 - 1.64
Emerald	1.570
Glass (varies with type)	1.5 -1.9
Topaz	1.610
Ruby	1.770
Saphire	1.770
Diamond	2.417
Iodine	3.340

Edge Transparency

Edge Transparency controls the visibility and therefore the definition of a *Transparent Object's* outer edges. You may wish to allow a *Transparent Object* to appear ghostly to its very edges. On the other hand, more realistic images of some *Objects* (like water drops) are obtained by giving the outer edges a greater visibility. The *Edge Transparency* function has three options: *Opaque*, *Normal* or *Transparent*. When *Opaque* or *Transparent* are selected, the *Edge Threshold* field becomes active.

Opaque, Normal, Transparent

Select one or other option as required for best visual results. This is usually found by trial and error.

Edge Threshold

The *Edge Threshold* value controls the visibility of *Opaque* or *Transparent Edge Polygons* with greater refinement. It determines the extent to which the *Surface* colour blends into the edge. This blending occurs in a zone, which may be wide and softly coloured, or narrow and sharply differentiated. The default value is 1.0. Higher values will soften the edge definition, whereas lower value will harden the edge definition. The best value is usually determined by trial and error.

Smoothing

Smoothing allows *Surfaces* to render smooth rather than be seen as faceted and polygonal. This is achieved through a mathematical routine called *Phong Shading*. Phong Shading utilises the *Maximum Smoothing Angle* (*MSA*) discussed in detail later (see Tutorial: *The Maximum Smoothing Angle - The Smoothing Threshold*). In most cases, you should enable *Smoothing* by clicking the button on (highlighted yellow). A *Smoothed* surface takes a little longer to render, but it is usually more realistic.

Max Smoothing Angle

Insert the desired MSA according to how the *Smoothing* routine needs to operate. Refer to the Tutorial: *The Maximum Smoothing Angle - The Smoothing Threshold*. This provides an insight to the *MSA* and a new and easier way of expressing it, the *Smoothing Threshold©.*

Double Sided

Click this button to convert all *Polygons* of the *Current Surface* into *Double Sided Polygons*. Refer to *Polygons* in the *Preamble* to understand the significance of *Double Sidedness*.

Bump Map

Click on the *Bump Map* button to invoke the *Texture Mapping* control panel, discussed later. Using an *Image* or *Procedural Texture* map, you can cause a *Surface* to appear 'bumpy' in accordance with the greyscale (luminance) values of the map. This is an important technique for generating natural-looking surfaces. You can, for example, cause a *Surface* to appear pock-marked or rippled. LightWave calculates how the lighting would affect such *Surfaces* and applies shading accordingly.

The Images Menu

Clicking the *Images* button pops up the *Images Editor* with which you control all aspects of *Image* use by Layout and Modeler. From here, you can load individual images, brushes, or sequences of images to be used as fixed backgrounds in *Scenes,* or fixed foregrounds in *Scenes*. An *Image* may be invisible within a *Scene* and used purely as a reflection in a *Surface*. An *Image* loaded here can be used as a *Backdrop Image* in Modeler (see Modeler/*Display/Options/Backdrop*). Any *Image* can be cleared from use, or replaced by another. The loaded *Images* may also be used for modifying the appearance of a *Surface* using *Texture Mapping* routines, discussed in more detail later. *Images* used here may have been created in LightWave or in external paint programmes and imported in a variety of file formats including RGB. *Images* or *Brushes* in .IFF format may be of any size and depth (2 to 24-bit), though there are limitations. LightWave can hold up to 1,000 *Images* in RAM, depending on how much is available. Insufficient RAM will be apparent from screen messages. For example, you may be warned of inadequate *Z-buffers* when a render is attempted.

Following is a description of the buttons and numeric data fields you'll find on the *Images Editor.* Most buttons have keyboard equivalents, which may be listed at any time by pressing the *Help* key.

Clear All Images

Clicking this button will *Clear All Images* currently held by the *Images Editor*. These include manually loaded *Images* and any which Layout has loaded automatically following the loading of a *Scene* which requires them (see *Scenes Editor*). Only *Images* held in RAM are cleared. *Images* stored in the *3D/Images* directory are not affected.

Load Image

Click here to *Load* an *Image* into the *Image Editor*, which makes it available for use in the *Current Scene. Images* cannot be used without assigning them a function within Layout or Modeler, though they may reside in memory until needed. The method of assigning an *Image* to a particular purpose depends upon the intended use. It may involve other control panels such as are found in the *Surfaces* menu, *Effects* menu and the *Texture*

Mapping system. When you click the button, the *Load Image File* requester pops up. This will search the default path for an *Image* file list . This is normally kept in the *3D/Images* directory. Click on the required *Image* file (or load one from another location) and click *OK* to complete the loading process.

Load Sequence

Load Sequence allows you to insert individual *Frames* of an animation or video sequence for use in the *Current Scene* (i.e. *Animation*). Only individual *Frames* may be loaded. You cannot load a compiled *.anim* file for example. *Image Sequences* can be used for the animation of *Surface Textures* in the form of *Image* maps, *Bump* maps, *Displacement* maps, *Clip* maps, etc.

Important Note

Before LightWave will accept an *Image Sequence*, the filenames of the individual *Images* must be in a recognisable form. Further, the *Image Sequence Prefix* requester requires exactly that, the file *Prefix*. LightWave records individual *Frames* of a rendered series using a three digit extension of the form '*Picture.001*'. This is an ideal format for use as an *Image Sequence.* Any sequence of *Images* you wish to use as an animation within the *Scene* (Rotoscoping) must have the name format '*Picture001*' or *'Picture.001'*. However, any other extension (.IFF, etc.) should be deleted, something which can prove to be a lengthy task! Further, to recognise that the *Image* files are part of a *Sequence*, each file must be named with a common root ('*Picture*' or '*Picture.*') followed directly by its sequence number (*001*, *002*, etc.) Accordingly, mixing *Picture.001* with *Pic.002* will not work, nor will *Picture.001* and *Picture002*. Only those *Images* which correspond with the *Prefix* name will be loaded. Also, the numeric part of the filename will be used to insert that *Image* into the corresponding *Frame* number of the *Current Animation.* If *Image* files are not numbered consecutively in the *Image* directory, then the 'gaps' in the current animation's list will be filled by the previous *Image* file. This will continue until the current *Frame* number and the next *Image* number coincide.

Clicking on the button pops up the *Image Sequence Prefix* file requester. The requester defaults to the *3D/Images* directory from which the first of a series of *Images* may be selected. Alternatively, redirect the search to an appropriate location. With a suitably named list of files (see above), select one bearing the *Prefix* name to be used (not necessarily the first *Image* to be used). In the filename field, delete the numeric notation so that only the *Prefix* appears. Click *OK*, or press *Return*, to load the *Image Sequence*. The *Images' Prefix* name will appear in the *Current Image* scroll bar, followed by the word *(sequence).* Only the first *Image* in *the Sequence* is loaded into RAM. The *Images* are individually loaded from the source directory as required by the rendering routine.

Images in Scene:

This information field indicates the total number of *Images* plus *Image Sequences* currently loaded into the *Image Editor.* It does not indicate the number of *Images* in any *Sequence*.

Total Image Memory Use:

This field indicates the total chip RAM consumption (in Kilobytes) due to the *Images* currently held in the *Images Editor*. This is the total size of all individual *Images* plus the first *Image* of any *Sequences*.

Current Image:

The *Current Image* scroll bar names the *Image* currently under *Image Editor* control. Click on the bar with the LMB to pop up a list of all presently loaded *Images* and *Image Sequences*. Select the required *Image/Sequence* by moving the cursor to the appropriate name and releasing the LMB.

Clear Image

Clicking this button will *Clear* the *Current Image* from the *Editor*. A warning will pop up asking for confirmation. Click on *Yes* or *No* as appropriate. *Images* stored on the hard drive are not affected.

Replace

The *Replace* scroll bar allows you replace the *Current Image* or *Image Sequence* with another. When you click and hold with the LMB, the option to replace with an *Image* or with an *Image Sequence* opens up. Place the cursor over the required option and release the LMB. The *Replacement Image File* or the *Replacement Image Sequence Prefix* requester will pop up. Both requesters default to the *3D/Images* directory. From here, or an alternative source, enter the name of the replacement *Image/Sequence* name and click *OK* to complete the replacement. You will be returned to the *Image Editor*, where the *Current Image* will now be the newly loaded replacement. All the assignments given to the original *Image/Sequence* will be automatically transferred to its

replacement. Read the *Important Note* under *Load Sequence* above, regarding the naming of *Image Sequences*.

Resolution: Bits/Pixel: Memory Use:

This information window provides numeric data on the *Current Image/Sequence*. Data on the first *Frame* of a *Sequence* is displayed.

Frame Offset

The Frame Offset determines which *Frame* number in an *Image Sequence* will be used in the *First Frame* of the *Current Scene* (*Animation*). Normally, the *Frames* of an *Image Sequence* are used in numeric order, starting with the lowest. The *Frame Offset* allows that order to be changed, or offset, by the number entered in the data field. The first *Image* used in an offset *Sequence* will be that numbered (O+F), where O is the *Offset* and F is the number of the *First Image* in the *Sequence*. The animated *Image Sequence* will loop from its last *Frame* to its first *Frame* during the course of the animation.

Loop Sequence

This button allows for the *Image Sequence* to *Loop* back to its starting point when the *Current Scene* (*Animation*) contains more *Frames* than the *Sequence*. After clicking on the *Loop Sequence* button, the *Sequence Loop Length* window is enabled (see below).

Sequence Loop Length

This data field defaults to 30, but can contain any number, which may be less than or greater than the total number of *Frames* in the *Image Sequence*. If you enter a smaller number, the *Image Sequence* will *Loop* after that number of *Frames* has been used in the *Current Scene* (*Animation*). Higher *Frames* will be ignored. To allow a complete *Sequence* to be used and then *Looped*, enter the total *Image* count. If the number entered is greater than the total *Image* count, the *Sequence* will run from the first to the last *Frame*. The last *Frame* will then be used until the stated *Loop Length* has been reached. If enough *Scene Frames* remain, then that process is repeated throughout the rest of the *Scene* (*Animation*).

Color Cycling

Images saved in .IFF format may be colour cycled by clicking on one of the *Color Cycling* activation buttons. *Color Cycling* is *Off* by default. *Color Cycling* is the creation of colour and motion effects in still *Images* or *Image Sequences*. This involves progressively shifting each colour in the *Image* palette to the next position as the *Scene* (Animation) *Frames* are rendered. The .IFF *Images* used should contain 256 colours or less and their files must be saved with their *Palette* data. Paint programmes use a colour table to allocate an *Index* number to each colour in the palette. You should keep a record of these *Indices* so that the required colour range can be cycled in LightWave. Their start and end positions are referred to as the *Low* and *High Cycle Indices* respectively. You can cycle through any contiguous block of colours up to the maximum (256).

Off, Forward, Reverse

Click on one or other button to activate *Color Cycling* in *Forward* or *Reverse* numerical order. Click *Off* to deactivate *Color Cycling*.

Low Cycle Index, High Cycle Index

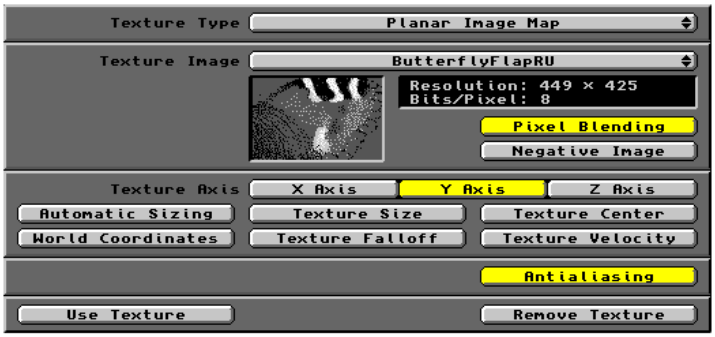
The *Low Cycle Index* is the number of the first colour to be used in the colour cycle. This would normally be set to the first colour in the *Image's* colour table. The *High Cycle Index* is the corresponding number of the last colour in the contiguous block you wish to cycle. It may be any number up to the maximum number in the palette. Thus, an 8-colour palette would be completely cycled by entering *Low* (0), *High* (7). Similarly, the last fifty colours in a 256 colour palette will be cycled by entering *Low* (205), *High* (255).

Continue

Click here to return to the Layout screen.

Texture Mapping

Clicking on any one of the seven *T* buttons in the *Surfaces Editor* pops up the *Texture Mapping* control panel shown below.



Using this panel, you can assign *Images* or *Brushes* (held in Chip RAM) as *Surface* coatings for *Objects* in the *Scene*. Their greyscale values may also be used in certain texturing routines, to modify another surface attribute. Alternatively, you may use a *Procedural Texture*, which is a computer-generated texture. These use a selected core algorithm which determines the broad texture type. Numeric data entered into pop-up windows then allow you to modify different characteristics of the *Texture*. In this way, an almost infinite variety of texture maps can be produced and applied. With these, you can modify the visual characteristics of any *Surface*. This may be in terms of its *Color*, its *Luminocity*, its *Diffuse* lighting, its *Specular* light emission, its *Reflectivity*, *Transparency* and roughness (*Bump* texture). The *Bump Map T* button in the *Surfaces Editor* pops a slightly more limited version of the *Texture Mapping* panel. Here, *Procedural Textures* are limited to *Ripples* and *Fractal Bumps.*

As well as modifying the *Surface* appearance, you can also use these textures to mould an *Object's* shape into that of the *Image* (*Displacement Mapping*) or you can cut away portions of the *Object* until its profile is like that of the *Image* (*Clip Mapping*). The *Texture Mapping* control panel is common to all these applications and its manipulation is roughly the same throughout. Note that the *Texture Mapped Surface* of an *Object* will retain the correct orientation when the *Object* is animated. However, this rule does not apply in the case of an activated *World Coordinates* button (see below). Following is a description of each button and data field associated with the panel.

Texture Type

Click on the *Texture Type* scroll bar to pop up a list of the types of texture available to you. There are fourteen types provided with the version 3.5 software. These are as follows:

Planar Image Map

Projects an *Image* onto a planar *Surface* to give it colour, or uses the *Image* as a *Clip Map*, *Bump Map*, *Displacement Map* or to modify another *Surface Attribute.* Best used where the *Surface* is reasonably flat. The procedure requires the use of an *Image* held in Chip RAM. It may be an *Image* or *Brush* produced in a paint package or other image processing software.

Several filetypes are recognised. *Images* and *Brushes* of .IFF type can be of any size and depth (2 to 24-bit).

Cylindrical Image Map

Allows the projection of an *Image* onto a cylindrical *Surface* to impart colour, or uses the *Image* as a *Clip Map*, *Bump Map*, *Displacement Map* or to modify another *Surface Attribute.* Use this on *Objects* which are broadly of cylindrical shape. Its *Image* requirements are as above.

Spherical Image Map

Allows the projection of an *Image* onto a spherical or spheroidal *Surface* to give it colour or uses the *Image* as a *Clip Map*, *Bump Map*, *Displacement Map* or to modify another *Surface Attribute.* Use this on *Objects* which are broadly spherical. *Image* requirements are as above. To map a sphere which will revolve (e.g. a planet) the *Image* should have matching edges so that the 'seam' is invisible. Note that projecting *Images* of planetary *Surfaces* may give better results using *Cylindrical* projection. This avoids *Image* squeezing at the poles and gives a more convincing render.

Cubic Image Map

Use this on *Objects* which are box shaped. A separate copy of the *Image* will be projected onto each face. You can also use the *Image* as a *Clip Map*, a *Bump Map* or to modify another *Surface Attribute.* The *Image* requirements are as above.

Front Projection Image Map

This system projects an *Image* from the *Camera's* viewpoint onto the *Current Surface*. Note that any *Image* projected in this way will line up perfectly with the same *Image* used as a *Background Image* (see *Effects Editor*). The *Front Projection* plus *Background Image* technique is useful when you want to make an *Object* slip 'into' the *Background Image*. Consider the two *Images* as a sandwich, into which you can move an *Object*. To do this, make a flat *Surface*, slotted, cut away or shaped as necessary. Onto this you apply the *Front Projection Image Map* and use the same *Image* as a *Background*. Place the flat *Surface* in such a position that your moving *Object* can slip through or behind it. Animate the *Scene* and the *Object* appears to slip into the *Background Image.* Easy!

Procedural Textures

Nine *Procedural Textures* are provided with v3.5 and are illustrated in a demonstration *Scene* called *TextureExamples.* The demo can be found in the *3D:Scenes* directory.

Checkerboard generates a squared design like a chess or checker board

Grid generates a three-dimensional grid across the *Surface*.

Dots produces a pattern of evenly spaced dots

Marble gives a *Surface* a marbled pattern in which veins are wrapped around a selectable axis.

Wood produces a wood-grain effect.

Underwater produces an undulating ripple across the Surface, similar to the surface of a water pool. Useful for a variety of applications, especially atmospheric effects.

Fractal Noise is a randomly generated 'noise' pattern and finds wide application for modifying many of the attributes you assign to *Surfaces*.

Bump Array is another fractal-based pattern which gives an irregular, bumpy appearance. Like all bump maps, this can have both positive and negative settings, providing indentations as well as raised areas.

Crust gives flaky, crusty look to *Surfaces* or to the attributes with which it is used.

When you select a *Procedural Texture* as your *Image Map*, a specialised sub-section of the *Texture Mapping* control panel pops up with which you can edit the *Texture* variables. The main buttons which appear on this panel are identical in function to analogous buttons on the *Texture Mapping* panel, described below. However, an additional set of buttons may appear which controls specific attributes of that *Texture*. A common feature throughout is the *Texture Color* button.

Texture Color

Clicking on this button pops up the *Texture Color Control* panel. This contains the familiar *Red*, *Green* and *Blue* component sliders plus a numeric data input field and colour swatch. Use these to set the colour you wish the *Texture* to exhibit when rendered onto the *Surface*.

Also appearing on the *Texture Variables* panel will be buttons which are specifically related to the *Texture* type. For example, the *Dots* texture has facilities for setting the *Dot Diameter* and its *Fuzzy Edge Width.* In general, these buttons are self-explanatory within the context of the Texture being edited. *Textures* which involve random or fractal calculations will require you to enter a *Frequency* setting and a *Turbulence* setting. The *Frequency* value is normally set between 1 and 6 and determines the number of core *Textures* being superimposed on the *Surface*. The *Turbulence* setting controls the vein or grain spacing incorporated into the calculation of the *Wood* or *Marble Texture*. Use *Turbulence* values of about half the *Vein*/*Ring Spacing* values. The *Ripple* type (found in *3D:Surfaces* and as a *Bump* map) as well as the *Underwater Texture* involve a *Wavelength* setting. This is self-explanatory, though the *Wave Speed* is slightly more complex. *Wave Speed* determines the rate at which waves propagate outwards during the animation. The value here should be *smaller* than the *Wavelength*. If they were identical, no movement of the *Texture* would be apparent because each *Frame* would show the next *Wave* exactly on top of the previous one. Use a value around a tenth of the *Wavelength*. The *Contrast* setting in the *Fractal Noise* panel controls the opacity of the *Texture* colour as it mingles with the base colour of the *Surface*. Use values up to about 4 for optimal results. As with many of LightWave's functions, experimentation is the soundest method of gaining an understanding of how they work and what will give the desired results. Experiment with the *Procedural Texture* settings, run a limited screen render on them and learn far more than any book will teach.

Texture Image

Click on the *Texture Image* scroll bar to pop up the available *Images* list. This list mimics that provided by the *Images Editor*, through which the *Texture Image* must be loaded. Move the cursor to the required *Image* and release the LMB. This selects an *Image* for use in an *Image Mapping* operation (see *Texture Type* above). The small screen displays a monochrome, low resolution thumbnail of this, the *Current Image Map*. Alongside the *Image* screen, the *Image Data Window* displays the *Resolution* (width x height) in pixels and the *Bits/Pixel* (pixel depth) of the *Current Image Map*.

Pixel Blending

Click on this button to allow LightWave to smooth out pixellation in *Bit Maps* which may be viewed at close range.

Negative Image

Many of the *Texture Mapping* routines employ the brightness element of the greyscale *Image* to modify the attribute to which it is assigned. The *Negative Image* button reverses the light and dark areas of the *Image* so that the mapping effect is flipped. For example, bumps become dips.

Texture Axis: X Axis, Y Axis, Z Axis

This set of buttons allows you to select which axis the *Current Image Map* will be projected along (*Planar*) or wrapped around (*Cylindrical* and *Spherical*). If the *Map* is to be the *Marble* or *Wood Procedural Texture*, these buttons determine the axis around which the veins/grain will be wrapped. Think of a column of *Marble* or a log of *Wood* to help with your selection. The other *Procedural Textures* do not require an *Axial* function.

Automatic Sizing

The *Automatic Sizing* button activates a mapping aid which automatically adjusts the *Texture Size* to that of the *Bounding Box* which encompasses the *Surface*. Further, it sets the *Texture Centre* (see below) to the *Centre* of the *Bounding Box.* When you click this button, a warning pops up advising of the loss of any *Texture Size* and *Centre values* that may exist. Click *Yes* or *No* as appropriate.

Texture Size

Clicking on this button pops up the *Texture Size* numeric input panel. These data set the *Size* of the *Current* (*Procedural*) *Texture*. The default value of 1.0 for the X, Y and Z axes indicate 1.0 unit of distance (metres) in all

directions. Scale the *Texture* by inserting appropriate values (usually smaller). Negative values will flip the *Texture* about that axis. Inserting values here does not limit the range over which the *Texture* will be applied, it simply controls the *Size* of the textural elements.

Texture Center

Clicking this button pops up the *Texture Centre* numeric input panel. The default settings of 0.0 throughout indicate that the *Current* (*Procedural*) *Texture* is centred at the *Origin* of the *Layout Grid.* You would normally enter the coordinates of the *Centre* of the *Surface* upon which the *Texture* will be applied, though this is not essential. A *Procedural Texture* is generated outwards from its own *Centre* and will eventually cover all locations in 3D space. Certain *Textures* (e.g.. *Checkerboard*) are automatically offset, so that their *Centre* will always be away from (0,0,0).

World Coordinates

This button is a toggle which allows you to 'anchor' the *Centre* of the *Texture* to the *Layout Grid*, i.e. the *World Coordinates*. This means that the *Texture* will not move with the *Surface* to which it is assigned. If the *Surface* is moved, the *Texture* will remain fixed on the *World Coordinates*. When animated, this gives the appearance of the *Object* moving 'through' the stationary *Texture Map*.

Texture Falloff

Clicking this button pops up the *Texture Amplitude Falloff (% per unit)* data input panel. With this you can control the rate at which a *Texture* 'falls off', i.e. becomes non-existent. The value you enter is a percentage which specifies how much fall off there will be for every unit of distance (metres) away from the *Texture Centre*. This can occur along any or all three axes.

Texture Velocity

Clicking this button pops up the *Texture Pattern Velocity (units per frame)* data input panel. With this you can move a *Texture* across a *Surface* at a determined rate. The default setting (0,0,0) indicates no change in the location of the Texture *Centre*. By inserting appropriate values in one or other field, the *Texture* will shift along the axes during the animation. The *Texture Velocity* is the rate of shift in units of distance (metres) per *Frame*.

Texture Amplitude

The *Texture Amplitude* data field allows you to set a value affecting the apparent 'height' (or 'depth') of *Bump Maps*. It determines the relative shading (contrast) applied between the illuminated and the non-illuminated sides of a bump or pit. The values are expressed in percentage, though may exceed 100% positive or negative. Use negative values for depressions which are mapped 'below' the general plane of the *Surface*.

Antialiasing

The *Antialiasing* button is a toggle which will reduce the artefacting of a finely detailed *Texture Map. Antialiasing* will help to reduce flicker, often observed when the detail exceeds the display resolution. This effect is especially noticeable when such *Images* are receding away from the *Camera*. Note that *Antialiasing* should be switched on via the *Camera Editor* (see below). *Antialiasing* increases rendering times considerably.

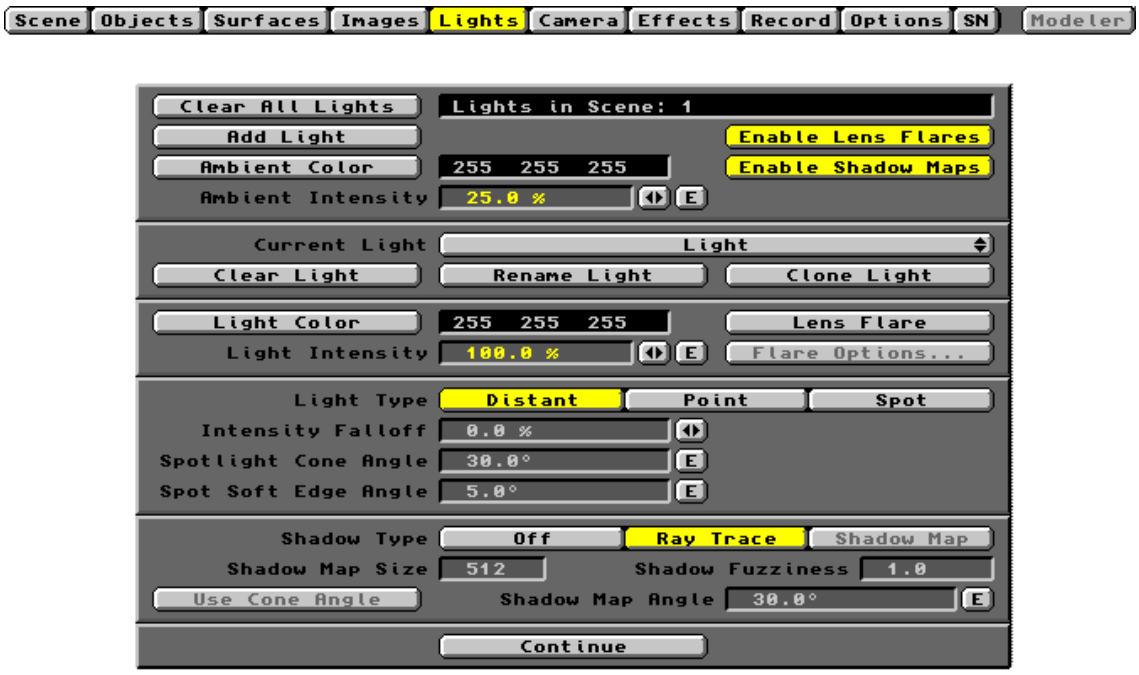
Use Texture

Click *Use Texture* to confirm your *Texture Mapping* settings and return to the *Surfaces Editor.*

Remove Texture

If you wish to *Remove* the *Current Texture* assignments, click the *Remove Texture* button. All the settings plus any *Texture Map* image will be removed from the control panel. This result can also be achieved while you are in the *Surfaces Editor*. To *Remove* any *Texture*, click the *T* button of the relevant *Texture*, while holding down the *s* key. After clicking this button, you will be returned to the *Surfaces Editor.*

The Lights Menu



The *Lights Editor* is accessed by clicking on *Lights* menu button at the top of the Layout screen. Using this, you have complete control over all aspects of lighting for the *Current Scene. Lights* can be adjusted to simply provide illumination or they may also cast *Shadows*. The *Shadows* may be hard-edged or soft-edged according to the nature of the *Scene* and the mood you wish to achieve. *Shadows* can be generated through a simple mapping routine, or they can be produced through the mathematically intensive system of ray tracing. Ray tracing is a slow process, but reproduces more closely what occurs in nature. *Lights* which you add to a *Scene* may be of three types and like many other components, they can be animated. Their brightness can be varied throughout the *Scene* (*Animation*) and you can arrange for them to display *Lens Flare* effects.

There are two prerequisites of *Lights* with which all *Scenes* must comply. A so-called *Ambient Light* is always present. This *Light* can be regarded as an all-pervading background light, without direction, but which provides some illumination for every *Surface* in the *Scene.* Though *Ambient Light* cannot be removed, you are able to control its *Intensity* and *Color*, so its effects are infinitely variable. The second prerequisite is the presence of one default *Light*, initially a *Distant* type, though you may change it to another type. Further than this, you may use as many or as few additional sources of illumination as you feel appropriate. The maximum number of *Lights* permitted in any *Scene* is 1,000, so a swarm of fireflies is a distinct possibility! However, to see any *Lights* in the

rendered image, you must apply *Lens Flare*. The phenomenon of *Lens Flare* and its relevance to *Light* visibiliy, is discussed in the Preamble.

Adequate lighting of a computer-generated *Scene* is relatively easy, but doing it well is an art. The subject of stage and set lighting is complex and one which can only be learned through experience. There are few rules, though you should endeavour to illuminate the players or their environment (or both) in a way that makes them fit together as a cohesive unit, the *Scene*. Contrast plays a vital part in many *Scenes* and is an important element in lighting control. An outstanding example of contrast in movie production is provided by Orson Welles' masterpiece 'Citizen Kane'. Also, the dramatic use of light to add atmosphere to a *Scene* is illustrated superbly by the interview of Rachel in Ridley Scott's 'Blade Runner'. Watch them both and learn.

Always avoid full face or front lighting, flooding the stage with light from a source positioned near the *Camera*. The best effects are achieved by using several low intensity *Lights* set in strategic places. In general, however, one *Light* should be dominant and this is typically placed above and to the side at about 45º. This looks best because this is how the sun works in illuminating the real world. Use shadows to create atmosphere in a *Scene*, especially soft-edged shadows which simulate what natural light produces. Alternatively, you can create drama through hard-edged shadows and by adding lens flares to any *Lights* which become visible to the *Camera*. Remember that the outward appearance of an *Object* is down to a combination of its *Surface* attributes and *Lights*. Both must work together to give you what you seek. Experimentation, trial and error and the determination to succeed are essential. They will ensure that the *Lights* and the *Surfaces Editor* together provide one of the most enjoyable experiences LightWave has to offer.
Following is a detailed description of every button and data field on the *Lights Editor* panel. Each button has a keyboard equivalent, a list of which can be accessed by pressing the *Help* key.

Clear All Lights

Click on this button to *Clear All Lights* which you have added to the *Current Scene*, i.e. those beyond the default *Light* and the *Ambient Light*. The default *Light* will return to its initial *Type* (*Distant*), *Name* (*Light*), RGB *Color* (255, 255, 255) and *Intensity* (100%). Its position and orientation will remain the same as they were in the *Scene*, or if animated to its default position (0,0,0) and orientation (60º, 30º, 0º). The *Ambient Light* returns to its default *Intensity* (25%) and RGB *Color* (255, 255, 255).

Add Light

Click here to *Add* another *Light* to the *Current Scene*. The *Light* added will be a default *Distant* type of 50% *Intensity* and white in *Color* (RGB 255,255,255). When you add another *Light*, its name (*Light*) will appear in the *Current Light* scroll bar (see below). The new *Light* will be located at the *Origin* of the Layout *Grid* (0,0,0) and orientated (0º, 0º, 0º). Move it into the required position by selecting it and either drag it with the mouse, or enter its required coordinates in *Numeric Data* fields associated with the *Move* and *Rotate* commands (see Layout *Mouse Group* above). The maximum number of *Lights* in any *Scene* is 1,000. Each additional *Light* adds considerably to the *Scene* rendering time.

Ambient Color

Click here to modify the RGB *Color* parameters of the *Ambient Light*. The *Ambient Color* adjuster panel pops up with which you can set the RGB values of the *Ambient Light*. The values (0-255) for the *Red*, *Green* and *Blue* component of the *Ambient Light* may be typed directly into the numeric fields using the keyboard. Each colour component is also determined by the position of its slide button. Click on a slide button and while holding down the LMB, drag it left or right to adjust that element of the colour. The values and the colour swatch beneath change as you move the sliders, so you can readily obtain the desired colour. The colour swatch can only display 4,096 different colours, though LightWave will calculate the exact colour from the numeric values set in the panel. This provides a true 24-bit, 16.8 million colour image. When you are happy with the colour click *OK*, otherwise click *Cancel*, to return to the *Lights Editor*.

Ambient Intensity

The *Ambient Intensity* window displays the brightness of the *Ambient Light* as a percentage of normal maximum. You can adjust this value in three ways. You can over-type the default value with the desired value using the keyboard, or you can change the setting by clicking on the adjuster button (**<>**) and dragging left (decreases) or right (increases) with the LMB, or you can apply an *Envelope* to animate the *Intensity* over several *Frames*. Clicking on the *E* button pops up the *Envelope Editor* panel, described in detail later. You can manually enter values greater than 100% if you wish the *Ambient Light* to be exceptionally intense. Similarly, larger values can be entered via the *Envelope Editor*, but the adjuster button only has a range of 0 -100%.

Lights in Scene:

The *Lights in Scene:* window indicates the total number of *Lights* (excluding the *Ambient Light*) in the *Current Scene*.

Enable Lens Flares

The *Enable Lens Flares* button toggles LightWave's lens flare capabilities on or off in the *Current Scene*. The default is on (button highlighted). When on, the *Lens Flare* and *Flare Options...* buttons are activated (see below).

Enable Shadow Maps

This button toggles LightWave's *Shadow Mapping* capability on or off in the *Current Scene*. The default is on (button highlighted, see below).

Current Light

The *Current Light* scroll bar displays the name of the currently selected *Light*. This is the *Light* whose properties are displayed in the data windows in the lower sections of the *Editor* panel. You can change the *Current Light* by clicking on the scroll bar with the LMB and moving the cursor to the desired name in the pop up list. Release the LMB to make this *Light* the *Current Light.*

Clear Light

Click this button to *Clear* the *Current Light* from the *Current Scene*. A warning panel will pop up asking for confirmation. Click *Yes* to confirm or *No* to cancel the operation.

Rename Light

Click this button when you wish to *Rename* the *Current Light*. The *Light Name* panel pops up, into which you should type an appropriate new *Name* using the Keyboard. When you are happy with the renaming, click *OK*, otherwise click *Cancel* to return to the Editor. The new name will appear in the *Current Light* scroll bar.

Clone Light

Click on this button to make one or more exact copies (*Clones*) of the *Current Light.* The *Number of Clones* panel pops up into which you may type the required number of *Clones*. The default number is 1, but you can make any number of *Clones* provided the total number of *Lights* in the *Sc*ene, excluding the *Ambient Light,* is no more than 1,000. Click *OK* or press the *Return* key to *Clone* the *Light*. A *Clone* inherits all the parameters invested in the original *Light*. These include its *Motion Path*, *Intensity Envelope* and *Lens Flare* settings. The *Current Light* name remains that of the original *Light*. Click on the *Current Light* scroll bar to pop up a new list of *Lights*.

Light Color

This button has an associated data window which displays the RGB colour components of the *Current Light's* colour. If you click on the *Light Color* button, the *Light Color* control panel pops up. With this you can adjust the colour of the *Current Light* by changing the *Red*, *Green* and *Blue* component values. This may be done by typing in the appropriate values (0-255) in the numeric fields using the keyboard. Alternatively, click on a slide button and drag with the LMB, left (to decrease) or right (to increase) the colour component value. The colour swatch below the slide buttons changes to the new colour. When you are happy with the colour settings, click *OK* to change the colour, otherwise click *Cancel*, to return to the *Editor*. The swatch can only display 4,096 colours, but LightWave will calculate the precise colour of the Light from the RGB values. This means a true 24-bit colour from a 16.8 million colour palette will be assigned and rendered.

Light Intensity

The *Light Intensity* window displays the brightness of the *Current Light* as a percentage of normal maximum. When you *Add* a *Light* to the *Scene*, it is automatically assigned the normal maximum *Intensity* of 100%. You can adjust this value in three ways. You can over-type the default value with the desired value using the keyboard, or you can change the setting by clicking on the adjuster button (**<>**) and dragging left (to decrease) or right (to increase) with the LMB, or you can apply an *Envelope* to animate the *Intensity* over several *Frames.* Clicking on the *E* button pops up the *Envelope Editor* panel, described in detail later. You can manually enter values greater than 100% if you wish the *Current Light* to be exceptionally intense. Similarly, larger values can be entered via the *Envelope Editor*, but the adjuster button only has a range of 0 -100%.

Lens Flare

The *Lens Flare* function allow you to simulate the well known phenomenon of camera lens reflection. The surfaces of glass lenses are usually subject to internal reflection when a bright light source is visible through the lens, but off-axis. The resulting multiple internal reflection can cause out of focus, ghost images of the *Light* source to appear on the photograph. Usually, these effects are avoided by film makers since they are not seen by the naked eye. However, flares are sometimes employed to add atmosphere to an image, especially if the *Camera* is looking towards a source of light. A further optical effect, which can be seen by eye, is an atmospheric refraction called 'starring'. Starring is the appearance of thin spikes of light radiating from the source, typically from stars on a clear night. LightWave's *Star Filter* system allows you to reproduce this effect

by generating an eight pointed spike effect around the source. Further refinements are possible, as described below. Clicking on the *Lens Flare* button enables the *Flare Options*... button. It has no effect on the intensity of the *Light* itself.

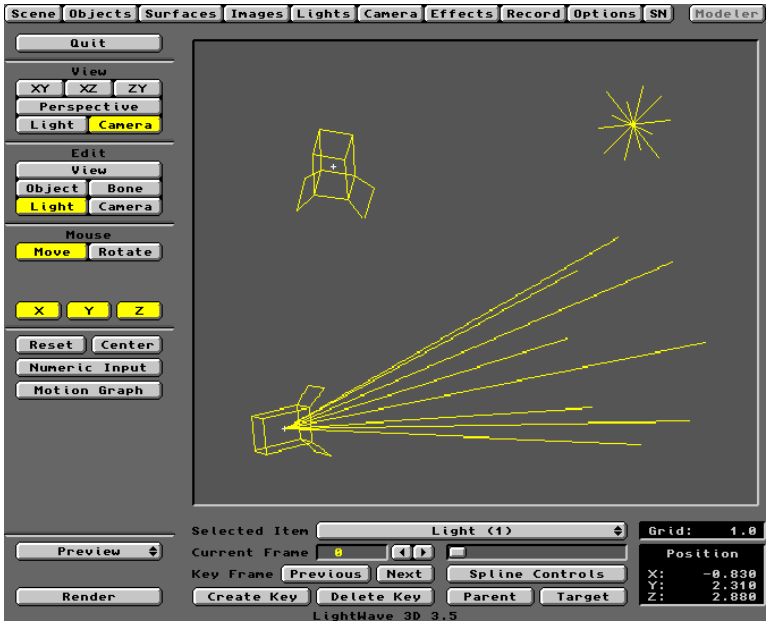
Flare Options...

Clicking on this button pops up the *Lens Flare Options for "Light"* Editor. This allows you to set up the optical effects of flaring and spiking discussed above. The *Lens Flare Options Editor* is described in detail at the end of this chapter.

Light Type

The *Lights* which you add to a *Scene* can be of three types, known as *Distant*, *Point* and *Spot* lights. The *Current Light* may be defined as one or other *Type* by clicking the appropriate button in the *Light Type* set. The *Default Light*, which is always present in a *Scene*, can be redefined in the same way. Each *Light Type* has a different icon in the Layout window, so you can visually identify the *Type* of the currently selected the *Light*.

Light Type Icons



Top left: *Distant Light* Top right: *Point Light* Bottom: *Spot Light*

Light Type / Distant

A *Distant Light* emits light in parallel rays, much like the sun. It can be likened to a light source placed at an infinite distance from the *Scene*. The intensity of a *Distant Light* is constant, irrespective of its location. Only the *Light* direction is important and this will change when you manipulate its orientation. The location of a *Distant Light* icon in the Layout window plays no part in its behaviour. Only the direction in which it points is important. It is useful to locate any *Distant Light* in the immediate area of the *Scene* simply to confirm its direction. The direction of the light emitted is clear from the icon, shown in the graphic below. The direction is also indicated in the information window at lower right of the Layout screen.

Light Type / Point

A *Point Light* emits its light radially in all directions. Accordingly, the orientation of a *Point Light* is irrelevant. Only its location affects the way its light interacts with other elements of the *Scene*. A *Point Light* may be likened to a candle flame or a bare electric light bulb, which can be placed anywhere in the *Scene.* Its intensity can be constant over distance, or you can apply a *Falloff* factor (see below) to vary its intensity. The *Point Light* icon is shown in the graphic.

Light Type / Spot

A *Spot Light* is the most regulated of the three types. It is very directional, emitting its light as a cone of rays. It can be likened to a car headlamp or a hand torch. The direction of the *Spot Light* is determined by its orientation. The *Spot Light* icon displays both its direction and light spread (cone angle). Its direction is also indicated in the information window at lower right of the Layout screen. The area of illumination provided by a *Spot Light* is controlled by its *Cone Angle* and also by its distance from the *Surfaces* upon which it shines.

Clicking on the *Spot* button activates the *Spot Light* control fields.

Point** (or) **Spot** / **Intensity Falloff

The *Intensity Falloff* is determined by the percentage reduction from normal maximum (100%) per unit distance from the source. The *Intensity Falloff* field allows you to insert an appropriate value. The distance from the *Light* at which the intensity is zero may be calculated as follows:

If: *Light* Intensity = *L* (%)
and: *Falloff* value = *F* (%)
Then: Distance of *Zero Intensity* = *L/F* units from the *Light*

To put this into context, a *Point Light* of 50% *Intensity* and a 10% *Falloff* will have *Zero Intensity* at 5 units from the source. Similarly, a *Spot Light of 85% Intensity* with a *5% Falloff* will have *Zero Intensity* at 17 units from the source.

Spot** / **Spot Light Cone Angle

The *Spot Light Cone Angle* is set by typing in an appropriate value in the numeric data field. The *Cone Angle* is the angle subtended by the axis of the *Light* and the outermost rays emanating from it. Therefore, the overall angular spread of the *Spot Light* is twice the *Cone Angle*. Values above 90° are not helpful. When a *Spot Light* is selected (or *Lights Visibility* is *On*), the angular spread of its light is indicated by 'beams'. These beams are visible from all viewpoints. From the *Light's* viewpoint, the spread appears as a circle to indicate its coverage. The *Cone Angle* can be animated by the use of an *Envelope*. Click on the *E* button to pop up the *Envelope Editor* (see below). Both constant and animated *Cone Angles* are visualised in the *Animation Preview*.

Spot** / **Spot Soft Edge Angle

The *Spot Soft Edge Angle* is set by typing in an appropriate value in the numeric data field. The *Cone Angle* of a *Spot Light* is controlled by a diaphragm in front of the lamp and its 'edge' can be given a hard or soft focus. Accordingly, the *Soft Edge Angle* varies the graduation between darkness and light when the beam is directed at a *Surface*. The *Soft Edge* is a annulus of graduated intensity around the outer edge of the illumination disk. The width of the annulus is determined by a second, smaller angled cone. The angle of this is set by the *Spot Soft Edge Angle*. Values above 90° are not helpful. This angle can also be animated by the use of an *Envelope*. Click on the *E* button to pop up the *Envelope Editor* (see below).

Shadow Type

The *Shadow Type* buttons give you three options for the type of shadows cast by the *Current Light.* However, every *Light* in the *Scene* (except *Ambient*) can be given any option, so you can obtain for a wide range of shadow effects. Note that for *Objects* to cast shadows, you must also set their shadow parameters in the *Objects Editor*. The *Shadow Type* options are described below.

Shadow Type** / **Off

Clicking the *Off* button switches off the shadow casting systems for the *Current Light*.

Shadow Type** / **Ray Trace

Clicking the *Ray Trace* button will cause LightWave to produce shadows using ray tracing routines. These simulate the way light behaves in the real world and the shadows produced are accurate, but hard-edged. Rendering times are significant due to the computations involved. To enable *Ray Tracing*, you must also select *Trace Shadows* in the *Camera Menu* (see below).

Shadow Type** / **Shadow Map

Shadow Mapping is only possible from *Spot Lights*. A *Shadow Map* is a simpler approach than *Ray Tracing*, so it is quicker to compute. The result is more natural in appearance because the shadows have soft edges. However, larger amounts of memory are required because the mapping method uses a profile map of each *Object* in the *Scene* which will be casting shadows. The smoothness of the resultant shadows is controlled by the amount of memory allocated to the process. This is determined by the *Shadow Map Size.*

Shadow Map Size

The *Shadow Map Size* field is activated when you click the *Shadow Map* button. The default value (512) may not provide enough RAM to generate the required shadow quality, which involves antialiasing routines. To increase RAM allocation, insert a larger value. The maximum is 1024, which produces idealised shadows at the expense of longer rendering times. The optimum value is best determined by trial and error for any particular system and *Scene.*

Shadow Fuzziness

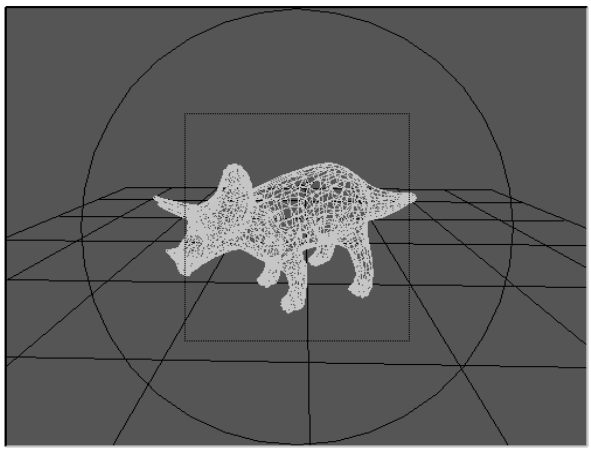
The *Shadow Fuzziness* field is also activated when you click the *Shadow Map* button. This value controls the shadow edge softness. A value of 0 produces a hard edged shadow (which may pixellate on large maps) while a value of 1 or more gives a progressive increase in edge *Fuzziness*.

Use Cone Angle

The *Use Cone Angle* button is activated when you click the *Shadow Map* button. Click it to use the *Cone Angle* of the *Current Spot Light* to generate the mapped shadow. The shadows produced will be more realistic than a simple mapped shadow.

Shadow Map Angle

The *Shadow Map Angle* field is activated when you click the *Shadow Map* button. It is inactive if you select the *Use Cone Angle* button. With the *Shadow Map* option selected, you can enter a *Shadow Map Angle* here to limit the area of mapping to a smaller section of the light cone. Values greater than the *Cone Angle* are not useful. The shadow is now rendered using a cone of light set by the *Shadow Map Angle* and not by the *Spot Light Cone Angle*. This is a useful strategy when RAM is limited. When you use this option, the *Light View* will show the shadow field. The circle denotes the area of illumination, while the dotted rectangle indicates the area which will be shadow mapped.



The Shadow Mapping Cursor

The *Shadow Map Angle* can be animated by the use of an *Envelope.* Clicking the *E* button pops up the *Envelope Editor* (see below).

Continue

Click here to return to the Layout display

Lens Flare Options Editor

This editor pops up when you click on the *Flare Options...* button in the *Lights Editor*. The buttons and data fields found on the panel are described below. All the buttons are toggles for the activation/deactivation of a *Lens Flare* effect.

Flare Intensity

You should enter a value in the field to set the brightness of the *Flare* for the *Current Light*. The *Intensity* of a *Flare* is independent of the intensity of the *Light* to which it is assigned. The intensity of the *Light* is not affected by the value given to its *Flare*. You can adjust this value in three ways. You can over-type the default value with the desired value using the keyboard, or you can change the setting by clicking on the adjuster button (**<>**) and dragging left (to decrease) or right (to increase) with the LMB, or you can apply an *Envelope* to animate the *Flare Intensity* over several *Frames*. Clicking on the *E* button pops up the *Envelope Editor* panel, described in detail later. You can manually enter values greater than 100% if you wish the *Lens Flare* to be exceptionally intense. Similarly, larger values can be entered via the *Envelope Editor*, but the adjuster button only has a range of 0 -100%.

Fade Off Screen

Highlighting this button will activate the *Fade Off Screen* function. This will cause the *Intensity* of the *Flare* to grow as it enters, or diminish as it leaves, the *Camera's* field of view. This edge effect is observed in movie camera lenses and may be used to add further realism to your animations. Click on the button again to deactivate the effect. Note that this function must be active if you intend to invoke the *Off Screen Streaks* effect (see below).

Fade Behind Objs

Highlight this button if you want the *Flare* to fade as the *Current Light* passes behind *Objects*. The effect is analogous to *Fade Off Screen* except it occurs near the edges of *Objects* which progressively obscure the *Light*. Transparent *Objects* will cause the *Flare* to be tinted by the colour of the *Surface* through which it passes. Click on the button again to deactivate the effect, in which case the *Flare* will not fade.

Fade In Fog

Activate the *Fade In Fog* button when you want the *Flare* intensity to diminish with distance due to a fog effect between the *Light* and the *Camera.* The *Fade* will be controlled by the *Minimum* and *Maximum Fog Distances* set up in the *Fog* control panel (see *Options Menu*). Beyond the *Maximum Fog Distance*, the *Flare* is invisible.

Fade With Distance

This button toggles the *Fade With Distance* function. When the button is highlighted, the *Flare* intensity will diminish as the distance between the *Camera* and the *Current Light* increases. This effect simulates the Inverse Square Law for all types of radiation. Thus, observed *Intensity* is inversely proportional to the square of its distance from the viewer. The *Flare Intensity* here is modified according to the *Nominal Distance* setting (see below).

Nominal Distance

This numeric data field is activated when the *Fade With Distance* function is switched on. The *Nominal Distance* setting is the distance (in metres) from the *Camera* at which the *Flare* has its nominal *Intensity*. The *Intensity* will vary inversely with the *Nominal Distance Factor*. This *Factor* is the multiplier which determines the actual distance between the *Current Light* and the *Camera*. Thus, a *Light* situated at *twice* the *Nominal Distance* will produce a *Flare* with *half* the nominal *Intensity*. At *half* the *Nominal Distance*, the *Flare* will be *twice* as bright.

Flare Dissolve

The *Flare Dissolve* value sets the transparency of the *Flare* effect. This allows you to visualise other effects which are otherwise swamped by the brightness of the *Flare*. For example, by *Dissolving* the *Flare*, you can enhance streaking effects (when set) without the intense central spot of the normal *Flare*. Enter the desired *Flare Dissolve* value in the numeric input field. You can do this in three ways. You can over-type the default value (0.0%) with the desired value using the keyboard, or you can change the setting by clicking on the adjuster button (**<>**) and dragging left (to decrease) or right (to increase) with the LMB, or you can apply an *Envelope* to animate the *Flare Dissolve* over several *Frames.* Clicking on the *E* button pops up the *Envelope Editor* panel, described in detail later. You cannot enter *Flare Dissolve* values greater than 100%.

Central Glow

This button toggles the appearance of a *Glow* of light placed at the source point of the *Light* . The *Glow* will have the colour of the *Current Light*. This button also toggles the activity of the *Red Outer Glow* button (see below). In order to render any *Point* or *Spot Light* which is visible to the *Camera*, the *Central Glow* button must be selected. All other *Flare* parameters require this glow to be on. Refer to the discussion of *Lights* and *Lens Flare* in the Preamble.

Red Outer Glow

This button toggles the appearance of a red, soft-edged halo around the *Current Light*. The effect is related to atmospheric interference. However, *Red* glows (i.e. long wavelength) are not always visible due to absorption by the intervening medium. Water, for example, absorbs the red component of light, so no *Red Outer Glow* is seen underwater.

Glow Behind Objs

This is not actually a lens effect at all, but is included here to complete the special effects of *Lights*. *Lights* produce a glow effect which is visible when the viewing medium contains fine suspended particles, such as fog, cloudy water, etc. Illumination of the medium may therefore be visible on the edge of an *Object* which is obstructing the *Light* itself. The *Glow Behind Objects* button toggles this effect. When you click on it, a warning pops up to advise you to use the effect for distant glows and not as a *Lens Flare* effect.

Central Ring

This button toggles the appearance of a small ring of light or halo around the *Current Light.*

Star Filter

The *Star Filter* button toggles the generation of refraction spiking around the *Current Light.* The effect produces an eight point spiking pattern which radiates from the centre of the *Light*.

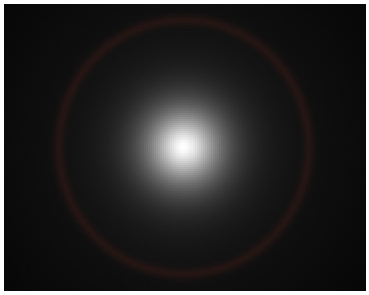
Random Streaks

This button toggles the generation of dozens of small streaks of light which radiate in all directions from the centre of the *Current Light*.

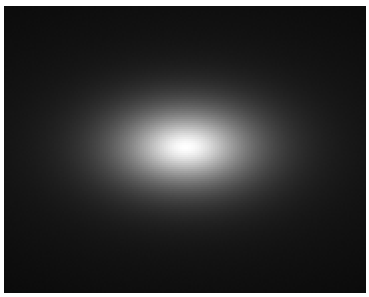
Off Screen Streaks

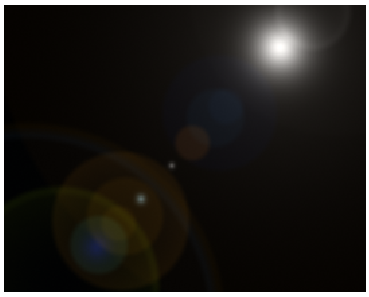
This button toggles the generation of *Random Streaks* radiating from the *Current Light* after it has moved beyond the Camera's field of view.

Anamorphic Squeeze



This toggles a modification of a *Lens Flare* effect, which squeezes it along the vertical (*Y*) axis. This gives it an elliptical form, making the *Flare* similar to those produced by anamorphic lenses.





Anamorphic Streaks

Toggles the generation of anamorphically squeezed streaks of light from the *Current Light.* The resulting blue streaks have an elliptical radiation field.

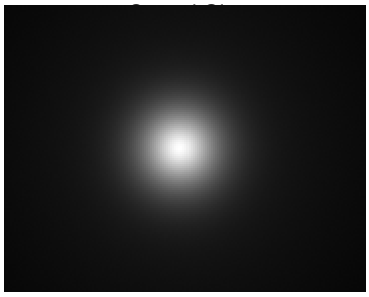
Lens Reflections

This button toggles the creation of *Lens Reflections*, translucent coaxial disks of light, which are produced by most standard, multi-element lenses. These *Flares* occur when the *Camera* lens is directed towards any intense *Light* source and produce reflections over a large portion of the image. The best effects are obtained when the Light is located towards a corner of the view. Animated *Lens Reflections* always move in a line which ends at the *Light*.

Continue

Click here to return to the *Lights Editor*

The following screen shots illustrate the the tremendous power of LightWave's Lens Flare facility.



Central Glow with Central Ring

Central Glow with Star Filter

Central Glow with Anamorphic Squeeze

Central Glow with Lens Reflections

The Camera Menu

Clicking on the *Camera* menu gives you access to the *Camera Editor*. This is where you set the parameters of the *Camera*, so that it will record the *Scene / Animation* in the format you require. The *Camera Editor* also contains switches to activate certain functions which are managed in other editors. The five sections of the *Camera Editor* allow you to control the quality and therefore the speed of rendering process, whether shadows will be generated and whether reflection and refraction will be integrated into the process. You can control the *Image* resolution and pixel aspect ratio as well as the boundary of the Layout screen which will be rendered. You can activate image quality factors like antialiasing and soft focus effects. The *Camera* type, the focal length of its lens and its focal ratio may be used to manage even more parameters, such as depth of field and motion blur.

The available controls allow you to produce test renders quickly, using all or just a small portion of the *Camera's* view. This allows you to test any aspect of your *Scene* without having to run a full render, saving considerable time. You have total control of the *Camera's* resolution, enabling you to provide images suitable for video recording, for film production or for printing. Advice of these and other aspects of image rendering is given at the end of the chapter. Each button and data field on the *Editor* is described below.

Rendering Type

Selects the type of rendering system in which the *Current Scene* will be produced. There are three options.

Rendering Type / ***Wireframe*** produces a wireframe rendition of the *Scene / Animation.* Analogous to *Preview* (also see Layout */Preview* button).

Rendering Type / ***Quickshade*** produces rendered *Images* without any shading / smoothing, texture mapping or transparency effects.

Rendering Type / ***Realistic*** gives the best image quality that can be achieved, using whatever shading, mapping and ray tracing options have been set. When the *Realistic* option is selected, three further *Ray Tracing* facilities become available.

Trace Shadows

This button switches on the *Camera's* involvement in the tracing of shadows. To produce a ray traced shadow in the rendered image, you must ensure that the following switches are also on:

Lights Editor: Ensure that *Ray Trace* is enabled for one or more *Lights*.
Objects Editor: Ensure appropriate *Objects* are able to *Cast*, *Receive* and/or cast *Self Shadows*

Trace Reflection

Click on this button to enable *Reflections* to be traced for *Surfaces* given a reflective value. *Single Point Polygons* will not render in a reflected image, nor will *Two Point Polygons* (lines), nor will *Objects* set to render in *Outline* format.

Trace Refraction

Click the *Tace Refraction* button to enable the rendering process to incorporate the *Refractive Index* (RI) setting of any transparent or semi-transparent *Surface* in the *Scene*. See also *Refractive Index* in the *Surfaces Editor*.

Basic Resolution

The *Basic Resolution* setting determines the height and width of the rendered image. These dimensions are measured in *Pixels*. The *Basic Resolution* affects visual quality and this may be augmented by other settings, such as the incorporation of an *Antialiasing* function. This and the *Adaptive Sampling* function, can make an image appear more finely resolved than its *Basic Resolution* would otherwise provide. The *Basic Resolution* scroll bar contains five resolutions at which you can render a LightWave *Scene*. Each option is referenced to the resolution usually required in video recording, i.e. *Medium Resolution (Video).*

Basic Resolution / ***Super Low Res (1/4 Video)*** is used for rapid test renders. The image resolution will be between 160 x 120 and 188 x 144 pixels, according to the *Pixel Aspect Ratio* used (see below). In all cases, the image will be blocky and pixellated, but useful information can be got quickly.

Basic Resolution / ***Low Resolution (1/2 Video)*** is also used for test renders, but provides a little more detail and therefore takes longer to produce. The image resolution will be between 320 x 240 and 376 x 288 pixels, according to the *Pixel Aspect Ratio* used (see below).

Basic Resolution / ***Medium Resolution (Video)*** is the option used in the production of video recordings where sharp images are required. The resolution will be between 640 x 480 and 752 x 576 pixels, according to the *Pixel Aspect Ratio* used (see below). The image quality may be improved further by the use of *Antialiasing* and *Adaptive Sampling*.

Basic Resolution / ***High Resolution (2 x Video)*** allows film and print quality images to be produced. The image resolution will be between 1280 x 960 and 1504 x 1152 pixels, according to the *Pixel Aspect Ratio* used (see below). This is LightWave's second highest resolution image rendering process.

Basic Resolution / ***Print Resolution 4 x Video)*** is the ultimate in image resolution and requires the longest time to render a *Scene.* The resolution will be between 2560 x 1920 and 3008 x 2304 pixels, according to the *Pixel Aspect Ratio* used (see below).

Custom Size

This button allows you to customise the rendered image size by entering a pixel *Width* and pixel *Height* value in the numeric data fields. The *Basic Resolution* settings are used for the *Custom Sized* image.

Custom Size / Width Enter the desired image *Width* in pixels.
Any value between 16 and 8000 is acceptable.

Custom Size / Height Enter the desired image *Height* in pixels.
Any value between 16 and 8000 is acceptable.

Overscan

Select *Overscan* when you want to render a broader image than the normal display provides. Overscanned images are needed in video recordings for television. The *Camera View* in Layout will indicate the *Overscan* area to be rendered.

Pixel Aspect Ratio

This is a scroll bar with five options, which change the shape of the image pixel according to the required output format. When you alter the *Pixel Aspect Ratio*, the Layout screen may show a dotted line frame. This is the boundary covered by the chosen format. Ensure that the images you wish to render are located within this boundary.

Pixel Aspect / D2 NTSC is the American *Video Toaster* standard.

Pixel Aspect / D1 NTSC is used for images which will be transferred to a D1 format recorder.

Pixel Aspect / Square Pixels is used for images you wish to use in Desk Top Publishing systems which need square pixels.

Pixel Aspect / D2 PAL is used in video recordings for European television.

Pixel Aspect / D1 PAL is used as above.

Letterbox

Select *Letterbox* to limit the dimensions of the rendered image to a letterbox shape. This effectively removes over half of the original image area by placing upper, lower, left and right boundaries on the Layout screen. A reduction of about one third will be made to the rendering time. The Layout screen will indicate the area to be rendered. It lies between the broken lines. Note that if *Letterbox* is activated under an *Overscan* rendering mode, the left and right sides of the letterboxed area will be expanded.

Limited Region

When you click on this button, you are able to control the area of the screen to be rendered. This is a time-saving facility that allows as much or as little of the screen to be rendered according to your need. It is especially useful when running tests on *Surfaces*, etc. After highlighting *Limited Region*, return to Layout and you may see a small rectangle drawn in a finely dotted line. In any case, hold down the ' l ' key to activate the *Limited Region* on the screen. It will turn yellow. While holding down the ' l ' key, use the mouse to drag or re-size the rectangle according to requirements. It is a little slow. When you are happy with the size and location of the *Limited Region*, release the LMB. The area will deactivate but remain visible as a fine grey outline. When you *Render* the *Scene*, only the area bounded by the *Limited Region* will be used. The *Limited Region* facility can be accessed whenever you are in Layout by simply pressing the ' l ' key. You do not need to invoke the *Camera Editor* to access it. However, you can only deactivate it by deselecting the *Limited Region* button on the *Editor*.

Segment Memory

Segment Memory is the amount of RAM allocated to image rendering. LightWave images are rendered from top down in horizontal slices called *Segments*. In this way, high resolution images can be rendered even when available RAM is small. The more RAM you allocate to *Segment Memory* the fewer the slices and the faster the image is produced. However, other functions require RAM to operate, so you must allocate it judiciously. Clicking the *Segment Memory* button pops up the *Segment Memory Size (bytes)* numeric data panel. Here, you can enter (numerically) as much memory as you feel is available. Remember the number entered is *bytes*, so a Megabyte of memory is 1000000. Click *OK* to set the allocation. The *Segment Memory* allocation and the corresponding number of *Segments* will appear in the information window (see below). You can estimate the memory required to render the image in one *Segment* as follows.

Required *Memory* (bytes) = Image *Height* (pixels) x Image *Width* (pixels) x 24

For complex *Scenes* involving a large number of *Objects*, *Images*, etc. you may not have the residual memory to render the image in a single slice, so reduce the allocation of *Segment Memory.* Over-allocation can result in LightWave being unable to perform its computations and may pop up a warning. Whenever this occurs, go back to the *Camera Editor* and reduce the allocation by half a Megabyte or so. In other circumstances, when *Segment Memory* is *just* insufficient, a process may be initiated but then crash. In this case the only way forward is a reboot, so be warned!

Resolution and Rendering Data Window

The window provides information about the current *Camera* resolution and the resultant image size, together with any *Limited Region* setting, *Pixel* and *Frame Aspect Ratios*, the *Segment Memory* allocated and the corresponding *Segments* count. These data are updated as you change the various settings.

Antialiasing

Antialiasing is an image processing technique which reduces the jagged (pixellated) appearance of *Surface* colour boundaries which have a diagonal aspect. The boundary is softened by the insertion of pixels with intermediate colours, according to the extent of the distortion. The *Antialiasing* button is a scroll bar with which you can select three degrees of image filtering (*Low*, *Medium* and *High Antialiasing*) or none at all (*Off*).

Antialiasing* / *Low is the quickest method involving five antialiasing passes.

Antialiasing* / *Medium provides a medium level of treatment involving nine passes.

Antialiasing* / *High gives the highest level of refinement, but is very slow since it requires seventeen passes through the antialiasing algorithm

Antialiasing* / *Off is the default setting and switches off the antialiasing process.

Soft Filter

This button is only active when *Antialiasing* is turned on. Clicking this button invokes a special image filtering algorithm which produces a softer edge appearance than simply antialiasing it. *Soft Filter* complements the *Antialiasing* setting to improve the typically 'clear cut' appearance of computed images. It can also be used as an alternative to *Motion Blur* (see below).

Adaptive Sampling

Clicking this button allows the *Antialiasing* process to incorporate an edge detection routine called *Adaptive Sampling*. This examines each slice of the image and compares the green colour component of adjacent pixels. The difference in their colour values is calculated and compared with a *Sampling Threshold*. Pixels which are below the *Sampling Threshold* are ignored, while those above it are antialiased to the degree of refinement set above. If the green components of adjacent pixels are less than eight units, then the red components are tested. The difference here must be twice the *Threshold* to invoke antialiasing. If red is below the *Threshold*, the blue components are compared. Here, the difference must be four times the *Sampling Threshold* to invoke antialiasing. The *Adaptive Sampling* button enables the *Sampling Threshold* to be adjusted.

Sampling Threshold

This numeric value sets the *Sampling Threshold*, that is the difference in green colour component for the pixel selection process described above. The permitted range for the *Threshold* is 0 (full antialiasing) to 256 (no antialiasing).

Film Size

The *Film Size* button enables LightWave to emulate a range of camera formats. The button is a scroll bar containing eleven different *Film Sizes*. In selecting a particular format, you alter the optics of the Layout *Camera*. This enables it to match the *Depth of Field* available to cameras using the selected *Film Size*. Thus, images rendered in LightWave may be incorporated into footage shot by those cameras. The default *Film Size* is 35mm motion picture format. When you click the scroll bar, a *Warning* message pops up to advise that the selection of a different *Film Size* affects only the equivalent focal length of the *Camera* and its *Depth of Field* effect.

Zoom Factor

The *Zoom Factor* allows the *Camera* to vary its field of view between that of a wide angle lens and a telephoto lens. Click in the numeric input field and type in a suitable value using the keyboard. Clicking on the *E* button invokes the *Envelope Editor* (see below) to allow the animation of the *Zoom Factor* setting.

Equivalent Lens

This information window indicates the *Equivalent* focal length of the *Camera Lens* used with the selected *Film Size* . The *Equivalent Lens* varies with the *Zoom Factor* setting. The *Equivalent Lens* and the *Film Size / Zoom Factor* affect the same *Camera* characteristic and which you use is a matter of personal preference.

Field Rendering

Field Rendering utilises the characteristics of the interlace system to allow two images for each *Frame* rather than one to be displayed. The technique is most useful when there is significant horizontal movement of *Objects* in the *Scene*. Without *Field Rendering*, large displacements near the *Camera* cause the *Object* to stutter across the field of view. *Field Rendering* inserts a 'half way' image between the nominal *Frames*, which are themselves split into alternate scan lines. The technique is especially useful for video work, but is less useful for other playback systems. If you try to display an individual *Frame* from a *Field Rendered* animation, it will seem to flicker as each 'half frame' is displayed alternately via the interlaced display. Running the animation, however, provides a much smoother transition from one *Frame* to the next. When *Field Rendering* is enabled, you also activate the *Reverse Fields* button.

Reverse Fields

This button toggles the order or field dominance of the interlaced images. This facility useful with video recorders requiring either odd or even line numbers to be recorded first. If interlaced half images are recorded with the wrong field dominance, a strobing effect is created when the animation is run.

Motion Blur

Motion Blur is the optical blurring of anything we perceive in rapid motion across our field of view. It does not occur when the eye follows the *Object.* To reduce *Motion Blur* in photography, you 'pan' the camera whilst operating the shutter. However, a good way to enhance the impression of speed is to actively cause the image to blur. This is achieved by fixing the camera's orientation and/or decreasing the shutter speed. If the speed of an object is high, blurring may occur even on movie film running at 25 *Frames* per second. Again, this phenomenon can be used to good effect in conveying an impression of speed. Clicking the *Motion Blur* button will cause any *Object* passing across the field of view of the *Camera* to be rendered as a blurred image. The blurring is calculated by comparing the position of the *Object* in consecutive *Frames* and applying a *Blur Length* factor. The *Blur Length* (see below) determines the 'length' of the blurring as a percentage of the *Object's* displacement per *Frame.* Using *Motion Blur* significantly increases rendering time.

Particle Blur

Click *Particle Blur* to cause blurring of rendered *Particles* (*Single Point Polygons*) crossing the field of view. *Single Point Polygons* may not render very well in an animation which requires their rapid transit across the screen. Use *Particle Blur* to convert *Point* images into streaks, which convey a better impression of movement. Clicking the *Particle Blur* button also activates the *Blur Length* field. Refer to *Motion Blur* above for a description of the *Blur Length*.

Blur Length

The *Blur Length* field is activated when you click on the *Motion Blur* button. Enter a numeric value (in percent) for the length of the blurring effect. The default is 50%, which simulates the blurring of typical movie cameras. You can insert the required figure using the keyboard, or by clicking and dragging on the adjuster button (**<>**), or you can invoke the *Envelope Editor* by clicking on the *E* button. The *Envelope Editor* (discussed below) allows you to animate changes in *Blur Length* as might occur during rapid changes in velocity.

Depth of Field

The *Depth of Field* (*DOF*) is a term photographers use to describe the range over which objects in the field of view are in sharp focus. All lenses have a *Depth of Field*, whose limits are located either side of the nominal focal plane. The size of the *DOF* depends on the focal ratio (F number) of the lens and on other factors. In photography, using the *DOF* effect adds to a sense of distance and depth, something which can be simulated in LightWave. Normally, all the *Objects* in a LightWave *Scene* are rendered in perfect focus no matter how far away, or how close they are located from the *Camera*. This can be changed by clicking on the *DOF* button, which activates the *Focal Distance* and the *Lens F-Stop* data fields. These allow you to control the *Depth of Field* of the Layout *Camera*.

Focal Distance

This is the nominal distance of perfect focus of the Camera. Any items at this distance from the Camera will be perfectly focused in the rendered image. The *Focal Distance* is measured in terms of LightWave units of distance (metres). Enter an appropriate value in the data field using the keyboard . Clicking on the *E* button

invokes the *Envelope Editor*, with which the *Focal Distance* of the *Camera* can be animated. You may wish to do this to create the impression of 'zooming into' the *Scene* with the *Camera*.

Lens F-Stop

The main factor which determines the *Depth of Field* (*DOF*) of a lens is its focal ratio, usually called the F-Number or F-Stop number. This is defined as the ratio of the focal length and the lens diameter (F/D). Since F is fixed, we must vary D to get different *DOF*s. This is achieved by blanking off the outer regions of the lens with an annular mask or 'stop', effectively reducing the lens diameter. By using an infinitely variable mask, all the useful range of F-numbers and therefore *DOF*s can be obtained. The *Lens F-Stop* field allows you to enter an appropriate F-Stop number for the Layout *Camera* . Conventionally, variable exposure camera lenses are calibrated in F-Stops which halve the exposure value from one Stop to the next higher. These settings are typically: F1.4, F1.7, F2.8, F4, F5.6, F8, F11, F16, F22, etc. As the *F-Stop* number increases, the *DOF* increases. Use the *Lens F-Stop* field to insert an appropriate value using the keyboard. You can animate the *DOF* by clicking on the *E* button to invoke the *Envelope Editor*, described later. The dramatic effects which can be achieved through the use of *Depth of Field* and *Lens F-Stop* settings are illustrated in the following graphic.

Depth of Field effect

Continue
Click here to return to Layout

Rendering Scenes - Some Advice

High resolution *Rendering* can be a very slow process and unless you are perfectly happy with the *Scene*, you should conduct some test renders at low resolution. This will speed up the testing process considerably and will avoid a lot of frustration. The use of the *Limited Region* (press the 'l' key) facility is also recommended for rectifying a particular part of the image, especially when you need to render it at a high resolution. Remember to deactivate this when you have finished testing (use the *Camera Editor*). Depending on what aspect of the *Scene* you wish to edit, the choice of the *Camera* resolution should be made according to the following.

To test Objects, Surface Colours and Lights

Set the *Camera Editor* to the following:

Rendering Type: *Quickshade*
Basic Resolution: *Low Resolution* or *Super Low Resolution*

To Test Surface Settings and Texture Maps

Set the *Camera Editor* to the following:

Rendering Type: *Realistic*

Basic Resolution: *Low* or *Medium Resolution*

To Reduce Rendering Time

Whatever resolution you wish to use, there are several ways that rendering times can be minimised during the preliminary testing. Apart from the *Limited Region* method, you could turn off any *Antialiasing* and *Adaptive*

Sampling settings. These processes are very time-consuming and may not be essential to the test. A further saving will be achieved by deactivating the *Trace Shadows*, *Trace Reflection* and *Trace Refraction* buttons.

Optimum Image Resolution

Depending on the intended use of the images, you should choose the optimum resolution. This will vary considerably, for example between the demands of a video recording and those of a printing system. The following are suggested as optimal for the uses given. However, there are no hard and fast rules and as with many aspects of 3D rendering, they are best determined through experience.

Video Applications

Set the *Camera Editor* to the following:

Rendering Type: *Realistic*
Basic Resolution: *Medium Resolution*
Antialiasing: *Low* to *Medium*
Adaptive Sampling: *8*

Film Applications

Set the *Camera Editor* to the following:

Rendering Type: *Realistic*
Basic Resolution: *High Resolution*
Antialiasing: *Medium*
Adaptive Sampling: *8*

Set the *Record Editor* to the following:

Save RGB Images: *On* (For subsequent output to film)

Print Applications

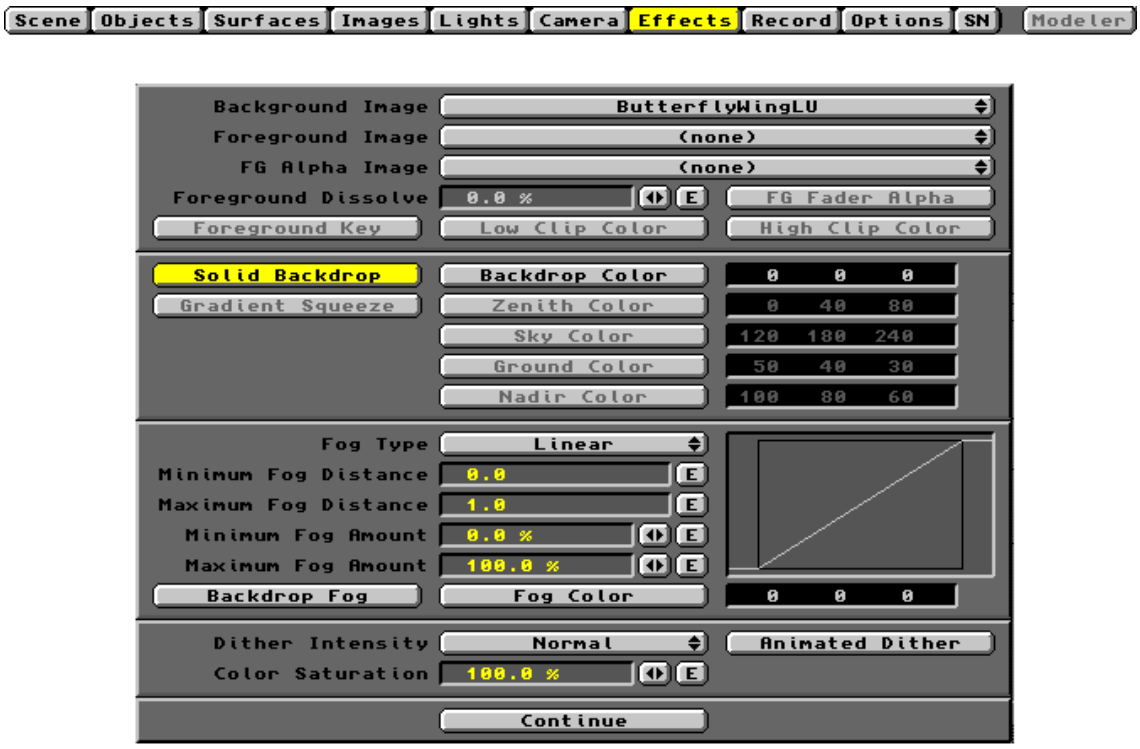
Set the *Camera Editor* to the following:

Rendering Type: *Realistic*
Basic Resolution: *Print Resolution*
Antialiasing: *High*
Adaptive Sampling: *8*

Set the *Record Editor* to the following:

Save RGB Images: *On* (For subsequent output to print)

The Effects Menu



Clicking on the *Effects* button pops up the *Effects Editor.* This is used for the management of background and foreground *Images*, to generate fade-ins or fade-outs and to create special effects through fogging. You can also modify the appearance of the render by the application of dithering routines. You can create 'keyhole' effects by *Alpha Channel* compositing and more. Each button and numeric data field on the *Editor* is discussed below.

Background Image

A *Background Image* (or *Image Sequence*) can be regarded as a fixed backdrop for the Layout *Scene*. It is located at an 'infinite' distance and always faces the *Camera*. It is always visible to the *Camera*, no matter what direction the *Camera* faces. It therefore behaves as if it were *Parented* by the *Camera*. Every *Object* you place on the stage will be seen in front of the *Background Image*, no matter how far away from the *Camera* they are located. Clicking on the *Background Image* scroll bar will provide a list of all currently loaded *Images* or *Image Sequences*. These are made available via the *Images Editor*, so you should ensure that all the required *Images* or *Sequences* are present. Refer to the *Images Editor* above for details on how to *Load Images* and *Image Sequences. Select the required one by holding down the LMB and moving the cursor to the appropriate name. Release the LMB to make that Image* the *Background Image.*

Note that the *Image* used as *Background* may be of any size, but to provide a full frame coverage it should be at least 752 x 480 pixels. This is analogous to *Medium Video Resolution* (see the *Camera Editor* for details). *Images* of a smaller size than this will appear with a border rendered in the *Backdrop* colour(s). The *Background Image* can be seen from the *Camera View* in Layout only when the *BG Image* button on the *Options Editor* is highlighted (see below).

Foreground Image

A *Foreground Image* is placed between the *Camera* and the Layout stage. It will therefore obscure any *Objects* on the stage, no matter how close to the *Camera* you place them. The *Foreground Image* is not visible in the Layout window, but will appear when the *Scene* is rendered. It will always render face on, no matter what direction the *Camera* is pointing. A typical use for a plain black *Foreground* is to be given a *Dissolve Envelope* (see below) to produce fade-ins or fade-outs. To cover the entire *Frame*, the *Foreground Image* should be at least 752 x 480 pixels in size (i.e. of *Medium Video Resolution*). See the *Camera Editor* for details about rendering *Images* of different *Sizes* and *Resolutions*. If the *Image* is smaller than this size, there will be a border around it when rendered. This border will be the edge portions of any suitably sized *Background Image* or it will be rendered in the colour(s) of the *Backdrop* (see below). Click on the *Foreground Image* scroll bar to pop up a list of all available *Images*. These must be *Loaded* via the *Images Editor* (see above). While holding down the LMB, position the cursor over the name of the required *Image*. Release the LMB to make the selected Image the *Foreground Image*.

FG Alpha Image

The *FG Alpha Image* button allows you to use a technique called *Alpha Image Compositing.* This image processing method enables two *Images* to be combined (composited) according to specific *Alpha Image* parameters. The two images can be any *Foreground Image* and any *Background Image*, or you could use a rendered *Frame* as the *Foreground Image* with any *Background Image.* An *Alpha Image* is a 256-level greyscale *map* which LightWave creates from any *Image* made available through the *Images Editor.* The luminance value of each pixel in the *Alpha Image* is used to generate a *Transparency Map*, enabling the seamless blending of the two other *Images*. The white areas of the *Alpha* map allow total coverage by the *Foreground Image.* None of the *Background* will be visible in these areas. Where the *Alpha* map is black, only the *Background Image* will be visible and the *Foreground* is transparent. This is like cutting soft edged 'key holes' in the *Foreground* through which the *Background* can be seen. Areas of grey in the *Alpha* map provide for subtle blending of both *Foreground* and *Background*, according to the *Alpha* luminance values.

A further advantage of the *Alpha* channel system is that any black areas in the *Foreground Image* will remain black after compositing. This means that the *Background* will not bleed through any areas of the *Foreground* which are black. Bleed through is a problem associated with simple luminance keying. *Alpha* channel compositing allows you to blend parts of different *Images* together as if they were actually rendered that way. For example, if you wanted to produce an image of the Mona Lisa with Cyrano de Bergerac's nose, this is what you'd do.

Place the image of the Mona Lisa as *Background*. Take a similarly sized image of Cyrano and with your favourite paint package, roughly outline his nasal features. Now paint the complete area outside it black. Reposition the nose within the frame in such a way that it will overlay Mona's nose in the *Background*. Now load the nose as the *Foreground Image* and also nominate it in the *FG Alpha Image* channel. Now render the combined images. The subtle shading around Cy's nose in the original image is used to blend the *Foreground* nose onto Leonardo's masterpiece. Wow! Also see Tutorial 12 for more in depth information on Alpha Channel Compositing.

Foreground Dissolve

Use *Foreground Dissolve* to render the *Foreground Image* more or less transparent. The numeric data field displays the degree of transparency as a percentage. When 100% *Dissolved*, the *Image* is invisible. At 0% *Dissolve*, the *Image* is completely opaque. This is a useful feature for creating fade-in and fade-out effects. Simply employ a black rectangle as the *Foreground Image*. Animate the *Dissolve* by applying an *Envelope* to the change its transparency. You can set any *Dissolve* value by entering the required number into the field using the keyboard, or click on the adjuster button (**<>**) and while holding down the LMB, drag left or right until the setting is correct. Release the LMB to accept the setting. To animate the *Foreground Dissolve*, click on the *E* button to pop up the *Envelope Editor* (see below).

FG Fader Alpha

The image compositing example given above can be achieved in a more subtle fashion using the *FG Fader Alpha* system. When you toggle this button on, the use of a *Foreground Image* rendered against or pasted to a black background becomes unnecessary. If you wish to composit a *Foreground* element (Cyrano's nose) on top of the *Background* (Mona Lisa's portrait), when the *Foreground* element (the nose) was not rendered over black (it was rendered on Cyrano's portrait), then you should use *FG Fader Alpha.* Here, those areas of the *Foreground* (Cyrano's portrait) corresponding to black areas of the *FG Fader Alpha* channel will be ignored.

Thus, if we load Cyrano's portrait as *Foreground* and *FG Alpha Image* and click the *FG Fader Alpha* button, Mo's image will dictate the general character of the composit, with Cy's bits visible according to their brightness. We really need Cy's nose to be highlighted and the rest of the portrait to be dark. Indeed, considering his silhouette, this is quite likely to be the case! The result may not be perfect because of colour differences between the two portraits. LightWave uses an additive routine to colour the final image, which might make the composit a different colour from what Leonardo painted. If colour distortion is observed, it can be minimised using the *Foreground Color Clip* buttons.

Foreground Key

Clicking on this button activates the *Low* and *High Clip Color* buttons. Using these, you can 'key out' a single colour, or a range of colours, from the *Foreground Image.* If you need to clip out a single colour, both *Clip Color* buttons should be set to that colour.

Low Clip Color

Click on this button to pop up the *Low Clip Color* panel. Use this to set the *Red*, *Green* and *Blue* colour components of the required *Low Color Clip*. The colour set here should be the darkest of the intended range. The panel contains numeric and slider methods to set the colour, which is displayed interactively on the swatch.

High Clip Color

Set the brightest colour of the intended range using the *High Clip Color* button. This pops up the *High Clip Color* panel. Adjust the *Red*, *Green* and *Blue* values of the required colour as noted above. When LightWave composites the *Images*, colours in the *Foreground* lying within the *Low* to *High Clip Color* range will be ignored and will not contribute to the composited image.

Solid Backdrop

The *Backdrop* is a 'built-in' *Background Image* in which the 'image' is simply a slab of colour. This provides a convenient and rapid way of framing the *Scene* in any plain colour. Use it for testing the arrangement of *Objects* and *Lights*, etc. Of course, there may be situations where a plain block of colour is required, for example when rendering animated intros or video titling. Clicking on the *Solid Backdrop* button toggles the activity of the *Backdrop Color* button.

Backdrop Color

Clicking on this button pops up the *Backdrop Color* panel with which you can set the colour of the *Solid Backdrop.* You can adjust the *Red*, *Green* and *Blue* components of the required colour by typing values into the numeric fields using the keyboard. Alternatively, you can use the slider buttons to adjust the colour, which is displayed interactively on the swatch. Click *OK* to accept the settings, which are then displayed in the adjacent window.

The *Backdrop* can be made to resemble a sky/land background by applying horizontal bands of appropriate colours. This is achieved using the following set of buttons.

Zenith Color

The *Zenith* is the area of sky immediately above your location. The Layout *Backdrop* behaves like a spherical envelope around the stage. It is therefore visible whatever the *Camera* angle. Furthermore, the *Camera* is always located at the centre of the *Backdrop* sphere, so the *Zenith* is always directly overhead. Clicking the *Zenith Color* button pops up the *Zenith Color* control panel with which you can adjust the colour of the 'sky' around the *Zenith*. Typically, a dark blue is used. Adjust the *Red*, *Green* and *Blue* values as described

Sky Color

The *Sky Color* button pops up the control panel for adjusting the colour of the 'sky' area of the *Backdrop*. The sky area covers the horizon to the *Zenith*. Typically a pale blue colour is used.

Ground Color

The *Ground Color* button pops up the colour control panel for adjusting the colour of the 'ground' area of the *Backdrop*. This area extends downwards from the horizon. Typically, a grey-green colour is used.

Nadir Color

The *Nadir* is the area directly below the *Camera* and diametrically opposite the *Zenith*. This button pops up the *Nadir Color* control panel with which you can select an appropriate colour. The typical colour used is an olive green.

Gradient Squeeze

Gradient Squeeze adjusts the extent of colour blending between the *Sky* and the *Zenith* areas, and between the *Ground* and the *Nadir* areas. When you click the button, the *Gradient Squeeze Values* panel is presented. Adjust the *Sky Squeeze* and *Ground Squeeze* values to appropriate settings. The higher the value set for *Sky Squeeze*, the closer to the horizon the colour blending will start. Similarly, the higher the *Ground Squeeze* setting, the closer to the horizon the *Ground* and *Nadir* colours will be blended. The default setting is 2.0 for each parameter.

Fog Type

The *Fog Type* button is a scroll bar providing you access to LightWave's fog management system. There are four *Fog Types* available. Select the required type using the mouse, while holding down the LMB. When the appropriate name is under the cursor, release the LMB. The fog generator places the *Camera* at the centre of a sphere of fog, whose density and distance from the *Camera* can be adjusted. Note that no matter how dense the fog is made, it will never fully obscure the *Backdrop*. Only *Objects* can be totally obscured. The *Fog Type* options are described below.

Fog Type / ***Off*** is the default setting. The *Fog* system is turned off.

Fog Type / ***Linear*** provides a fog density which is constant from 'front' to 'back'. The visibility of *Objects* in the fog varies directly with their distance from the *Camera*. The optical density of the fog is represented graphically in the window to the right of the *Fog Type* button. When any *Fog Type* is selected, the four data fields immediately below the button are activated. These are discussed after the *FogTypes* have been described.

Fog Type / ***Nonlinear 1*** provides a non-linear density distribution which is readily appreciated from the optical density graph. *Objects* moving away from the *Camera* appear to disappear in a more natural way with a non-linear setting.

Fog Type / ***Nonlinear 2*** is a more acute non-linear density distribution in which *Objects* disappear more rapidly as they enter the fog.

Minimum Fog Distance

This numeric setting controls the distance the front edge of the fog is from the *Camera* . Between the *Camera* and this distance, all *Objects* are seen normally. The setting is calibrated in standard units of distance (metres). Click the *E* button if you wish to animate the *Minimum Fog Distance* by the use of an *Envelope* (see below).

Maximum Fog Distance

This setting determines the distance from the *Camera* at which *Objects* in the fog are at their least visible. Total invisibility will occur only if the *Maximum Fog Amount* is 100% (see below). In that case, only fog will be seen beyond the *Maximum Fog Distance*. Click the *E* button if you wish to animate the *Maximum Fog Distance* by the use of an *Envelope* (see below).

Minimum Fog Amount

This setting determines the relative density of the fog at its nearest point to the *Camera*. A setting greater then 0% will create a 'wall' of fog standing at the *Minimum Distance*. Type in the required setting, or use the adjuster button (**<>**). The graphical display will update to the new setting. Click the *E* button if you wish to animate the *Minimum Fog Amount* by the use of an *Envelope* (see below).

Maximum Fog Amount

This setting determines the relative density of the fog at the *Maximum Distance* from the *Camera*. Settings below 100% will permit *Object* visibility beyond the *Maximum Distance* of the fog. Type in the required setting, or use the adjuster button (**<>**). The graphical display will update to the new setting. Click the *E* button if you wish to animate the *Maximum Fog Amount* by the use of an *Envelope* (see below).

Fog Color

Click on the Fog Color button to pop up its adjuster panel. With this you can adjust the *Red*, *Green* and *Blue* components of the required colour by typing values into the numeric fields using the keyboard. Alternatively, you can use the slider buttons to adjust the colour, which is displayed interactively on the swatch. Click *OK* to accept the settings, which are then displayed in the adjacent window. A typical *Fog Color* setting is (150,150,160), which is most suitable when the *Backdrop Color* settings are similar value. However, because the fog never totally obscures the *Backdrop*, a better method is to use *Backdrop Fog* colour.

Backdrop Fog

More realistic colour gradation is produced when you select *Backdrop Fog* to assign the *Fog Colour*. Clicking this button deactivates the *Fog Color* button. Using *Backdrop Fog* assigns its colours to the fog itself. This causes objects to blend into the *Backdrop Colors* rather than into a single *Fog Color* plus a residue of the *Backdrop Colours.*

Dither Intensity

Dithering allows coloured pixel mixing to generate intermediate colours and so avoid banding. Use LightWave's fog dithering system to achieve interesting effects from colour blending of *Objects* and *Fog Color* or *Objects* and *Backdrop Fog* colours. The *Dither Intensity* button is a scroll bar which provides our options when clicked.

Dither Intensity* / *Off means the *Dithering* system is switched off. Some colour banding may be seen when fog and *Object* colours are blended.

Dither Intensity* / *Normal provides the first level of dithering available. Banding should be virtually absent.

Dither Intensity* / *2 x Normal increases the dither routine even further and can be used for outputting to high resolution recording or printing systems.

Dither Intensity* / *4 x Normal gives the maximum available dithering, which will cause the rendered image to look grainy. This gives a look similar to fast photographic or movie film. Where graininess is the desired result, it can be particularly effective when animated (see below).

Animated Dither

The distribution of colour pixels during the dithering routine can be varied so that each *Frame* has a slightly different arrangement. This is *Animated Dithering* and makes a *Surface* appear to shimmer or sparkle. The routine can only be operated when the *Dither Intensity* is *Normal* or higher.

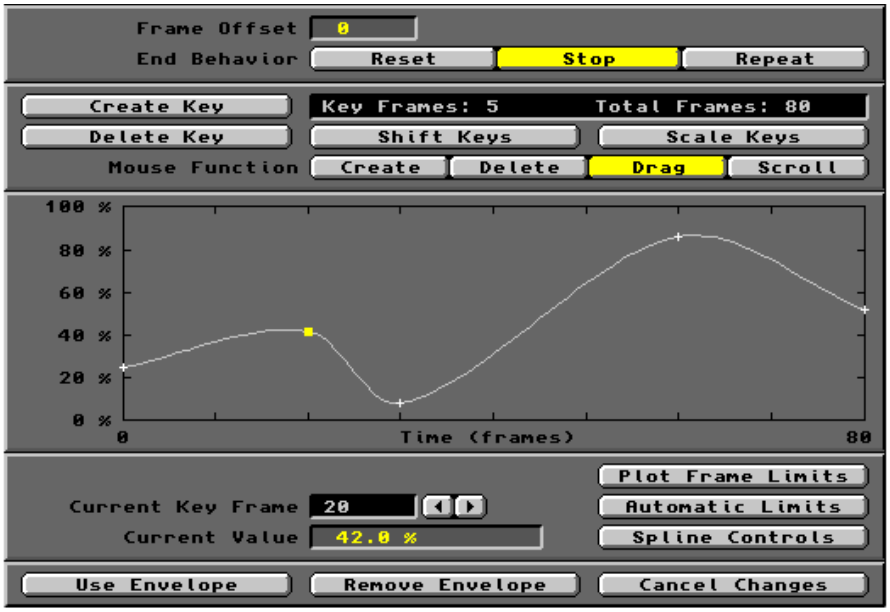
Color Saturation

Use this numeric field to set the amount of colour in the *Scene*. Fully saturated colours have a 100% setting. Values less than 100% cause the colours to fade to their greyscale values. At 0% saturation, the *Scene* will be rendered in 'black and white'. Insert the desired *Color Saturation* value using the keyboard, or using the adjuster button (**<>**). If you wish to animate a change in *Color Saturation*, click on the *E* button to pop up the *Envelope Editor* (see below).

Continue

Click here to return to Layout.

The Envelope Editor



Whenever you click on an *E* button in one of the main menu *Editors*, you invoke the *Envelope Editor*. In LightWave version 3.5, this *Editor* is common throughout all applications and doesn't contain any reference to any particular activity. Thus, the *Envelope Editor* accessed from the *Objects Editor* is identical to that associated with the *Effects Editor* and so on. Later editions display the name of the effect you are editing, otherwise the principles are identical. The *Envelope Editor* is used in the animation of special effects. Examples include the *Metamorphing* of one *Object* into another *Object*, the dimming of *Lights* and the creation of fade-out. Making animations look good depends on the use of the *Envelope Editor.*

An *Envelope* describes the change in a specific element of the *Scene* throughout a series of *Frames*. The *Envelope* contains all the data needed to bring those changes about. Since the *Frames* of an animation are displayed at a regular rate, we can equate the *Frame* number with *Time* from the start of the animation. The *Envelope Editor* not only describes a change, it allows you to alter the rate at which that change occurs. As with the Layout interface, the *Envelope Editor* uses its own *Key Frames*. These are specifically associated with the effect to which they relate. They are not affected, nor do they affect, any of the *Key Frames* which you create in Layout. As with *Motions*, the use of *Splines* is available to the *Envelopes Editor,* so you can refine the changes you are designing.

The changes in any particular element over time are presented graphically as the *Envelope Graph.* This can be edited using both keyboard and the mouse. The *Envelope Graph* is created by joining points which represent the numeric value of the element at a specific *Time (frame)*. It consists of a white line, or curve. The *Graph* display is referred to as the *Current Envelope*. At the start and end of the curve are white crosses (+), known as *Key Points*, indicating the first and last *Key Frames* in the *Envelope*. The currently selected *Key Point* is indicated by a yellow block. It will be referred to herein as the *Current Key.* Intermediate crosses represent intermediate *Key*

Points. The horizontal axis of the graph is always *Time* (*frames*). The vertical axis is the numerical value of the effect at that stage. This may be just a number or it may be expressed as a percentage. The vertical axis is automatically calibrated against the effect you have nominated via the *E* button. So, if the effect is expressed as a percentage of maximum, the axis will be calibrated in percent. The command buttons found on the *Envelope Editor* panel are described below. Keyboard shortcuts for various actions are displayed by pressing the *Help* key whenever the *Envelope Editor* is displayed.

Clear Envelope

Click on this button to *Clear* the *Current Envelope* from memory. You will not affect any *Envelopes* stored on the hard drive. The button simply clears the data held in RAM.

Load Envelope

Click this button to *Load* an *Envelope* held on the hard drive and assign it to the *Current Scene* element. *Envelopes* are stored in the *3D:Envelopes* directory. The *Load Envelope File* requester pops up. This will search the default path (*3D:Envelopes)* and provide a file list of existing *Envelopes*. Click on the required filename and then on *OK* to *Load* the *Envelope* into RAM. Any difference between the number of *Frames* covered by this *Envelope* and the number of *Frames* in *Current Scene* will need to be addressed. This will involve using a *Frame Offset* (see below) to adjust the timing of the *Envelope* to coincide correctly with the event in *Scene*.

Save Envelope

After editing an *Envelope*, you can *Save* the data in the *3D:Envelopes* directory. Click on this button to *Save* the *Current Envelope* file. The *Save Envelope File* requester pops up. This defaults to the above directory. Enter the filename under which you wish to *Save* the *Envelope*. Click on *OK* to complete the saving and return to the *Envelope Editor*.

Frame Offset

If you need to delay the effect defined by the *Current Envelope* to a later phase of the animation, use this field to enter the *Frame Offset*. This number is the number of *Frames* which you wish to 'skip' before the event starts. This could used when a *Loaded Envelope* covers a smaller number of *Frames* than there are in the *Current Scene*. Note that *Frame* 0 of the *Envelope* remains at *Frame* 0 of the animation. Only the subsequent events are *Offset* by the number of *Frames* entered in the data field. The *End Behavior* (sic!) settings are *Offset* to the same degree.

End Behavior

This determines the course of an event upon reaching the end of the *Current Envelope.* There are three options.

End Behavior* / *Reset will reset all *Envelope* parameters to those pertaining after the final Key Frame. In most cases, these will be their default settings.

End Behavior* / *Stop will cause the event to remain in the condition set by the last *Key Frame* of the *Envelope*.

End Behavior* / *Repeat causes the *Envelope* to be repeated until there are no further *Frames* available. This is a useful setting for an event you wish to continue throughout the animation. A good example of a *Repeat* effect would be a butterfly scene in which its flapping wings are controlled by a two-*Frame* morph. *Frame* 0 has the wings up, *Frame* 1 has the wings down. The *Metamorph Envelope* for this is a simple 0-100% change covering two *Frames*. With the *Repeat* option set, this *Morph* will continue throughout the animation, no matter what its length.

Create Key

Click here to *Create* a *Key Frame*. The *Create Key at Frame* panel pops up in which you should type the intended *Key Frame* number. Click *OK* to confirm the action. When you have *Created* the *Key*, adjust the value of the *Key* using the mouse or via the keyboard (see *Mouse Function* and *Current Value* below).
The data window to the right of the *Create Key* button displays the *Number* of *Key Frames* and the last *Frame* number of the *Current Envelope*.

Delete Key

Click this button to *Delete* a *Key Frame* from the *Current Envelope*. The *Delete Key at Frame* panel pops up. This will default to the *Current Key* number, which can be edited using the keyboard. When you have entered the appropriate *Key Frame* number, click *OK* to complete the *Deletion*. The *Envelope Graph* will update accordingly. Note that *Frame* 0 cannot be deleted.

Shift Keys

Use the *Shift Keys* button to shift all or a selection of the *Keys* to a higher or lower *Frame* number. This advances or delays the effect by a period equivalent to the *Shift* setting. Clicking on the button pops up the *Shift Keys* panel, which contains four numeric data fields. Enter the relevant data using the keyboard.

Shift Keys* / *Low Frame is the number of the first *Key* in the range to be *Shifted*.

Shift Keys* / *High Frame is the number of the last *Frame* in the range to be *Shifted*.

Shift Keys* / *Shift Frames By is the number of *Frames* the specified range is to be *Shifted* by.

Shift Keys* / *Shift Values By is the numeric value to be added to, or subtracted from, each value in the specified range. Positive values are added, negative values are subtracted.

Scale Keys

Scale Keys allows you to expand or contract the *Current Envelope* so that its effect occurs more slowly or more quickly. This is achieved by fitting the changes contained in the *Current Envelope* into a larger (slows) or smaller (quickens) number of *Frames*. Clicking the button pops up the *Scale Keys* panel, which contains four numeric data fields.

Scale Keys* / *Low Frame is the number of the first *Key* in the range to be *Scaled*.
Scale Keys /High Frame is the number of the last *Key* in the range to be *Scaled*.

Scale Keys /Scale Frames By is the multiple you wish to *Scale* the length of the effect by. For example, a value of 5 will increase the duration of the effect five-fold. The total number of *Frames* will increase five-fold and the position of the *Keys* will be adjusted to provide the required event profile.

Scale Keys /Scale Values By is the multiple you wish to *Scale* all *Key* values by. Numbers greater than 1.0 will enlarge the existing values. Numbers less than 1.0 will reduce the existing values

Mouse Function

The *Envelope Graph* can be edited directly using the mouse. The *Mouse Function* buttons assign different functionalities to the mouse, which can then be used by clicking, or clicking and dragging the cursor within the *Graph* section of the *Editor* panel. To assign one of the functions, click the relevant button. Only one button at a time can be highlighted. Note that your editing of the *Graph* invokes automatic *Spline* adjustment of the curve. As you change the coordinates of *Key Frames*, LightWave will calculate the optimum *Spline Curve*. You can change this later using the *Spline Controls* button (see below).

Mouse Function* / *Create assigns *Key Frame Creation* ability. Place the cursor at the required coordinates on the *Graph* and click with the LMB. A new *Key* is created at that point. You should place the cursor as near as possible to the required *Frame* location. *Create* has 'snap to grid' functionality and will snap to the nearest available *Frame* number. If a *Key* already exists at that position, a *Error* message will pop up. The newly *Created Key* becomes the *Current Key.* The *Graph* updates interactively.

Mouse Function* / *Delete assigns the ability to *Delete Key Frames.* Place the cursor on the *Key* you wish to remove and click with the LMB. The *Key* is made the *Current Key* and immediately *Deleted*. The next higher *Key* is made the *Current Key Frame*. The *Graph* updates interactively. You cannot *Delete* the first *Key Frame*.

Mouse Function* / *Drag assigns *Drag* functionality according to which mouse button is used. Hold down the LMB to drag a *Key Frame* to a new *Value*. If the *Value* is a percentage, there may be little point in exceeding 100% up or down, even though the mouse will drag beyond those values. Hold down the RMB to *Drag* a *Key* along the *Time (frames)* axis. You may *Drag* a *Key* until it is adjacent to the next lower or the next higher *Key Frame*. The *Graph* updates interactively. You cannot *Drag* the first *Key Frame* from its position at *Frame* 0.

Mouse Function* / *Scroll assigns *Scrolling* ability to the mouse. This allows you to scroll the *Graph* left or right to locate a position beyond the currently displayed range. The displayed range of the *Graph* is normally sixty *Frames.* If you wish to see beyond the *Frame* 60, click the cursor anywhere on the *Graph* and holding down the LMB, drag it to the left. The *Graph* will scroll to the left, revealing a new area where you can *Create* further *Keys* as appropriate. Drag to the right to return to the original area. If you generate a *Graph* which exceeds sixty *Frames*, the *Scroll* function will be required to access all the *Keys*. (Also see *Automatic Limits* below).

Current Key Frame

This window displays the number of the currently selected *Key Frame*. You can skip between *Keys* by clicking on the left or right skip button. The *Current Key* indicator will skip to the selected *Key Frame* number.

Current Value

This window displays the *Value* assigned to the *Current Key Frame*. You can alter this by typing in the required *Value* using the keyboard. The *Graph* will update interactively.

Plot Frame Limits

Click this button to pop up the *Plot Frame Limits* panel. Here, you can enter (keyboard) the lower and upper *Frame* numbers for the *Time (frames)* axis. The *Graph* will be limited to those *Frames*. This facility allows you to 'zoom' into a small section of the *Graph*, or to display the whole contents of an extensive *Envelope*.

Automatic Limits

The *Automatic Limits* button will automatically fit an entire *Graph* within the display area, irrespective of the number of *Key Frames* it contains.

Spline Controls

Clicking the *Splines Controls* button pops up the *Current Key Frame Spline Controls* panel. For details on this editor and on *Splines* in general, see the description of Layout's *Spline Controls* below.

Use Envelope

Click the *Use Envelope* button when your editing of the *Graph* is complete and you wish to incorporate it into the current *Scene*. This command replaces the data in RAM with the new *Envelope* file. The *Envelope Editor* is closed and you are returned to the main menu *Editor* from which invoked the *Envelope Editor*. The *Scene* in RAM will incorporate the new *Envelope*.

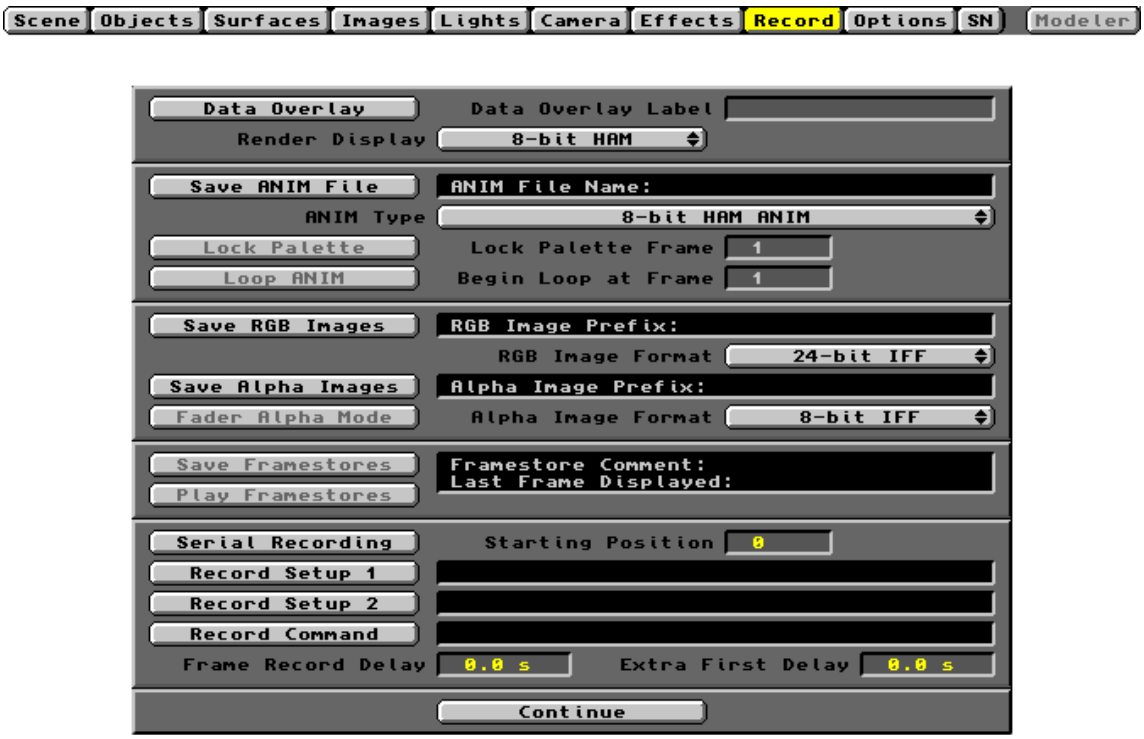
Remove Envelope

Click this button to *Remove* the *Current Envelope* from RAM. Only the *Scene* held in RAM is affected. The *Scene/Envelope* files stored on the hard drive are not affected, unless you *Save* the *Current Scene/Envelope*. In this case the hard drive files will be updated. You will be returned to the main menu *Editor* from which invoked the *Envelope Editor*.

Cancel Changes

Click this button to *Cancel* any *Changes* you have made to the *Current Envelope* that have not already been saved. The *Envelope* held in RAM will remain in its as-loaded state. You will be returned to the main menu *Editor* from which invoked the *Envelope Editor*.

The Record Menu



The *Record Editor* is pops up when you click on the *Record* menu button. Using this, you can control all aspects of *Animation* file creation and recording. The dimensions and resolutions of the *Frame* renders are controlled by the *Camera Editor* (see above). The progress of the render can be monitored by selecting a *Render Display* setting. The files are rendered and saved in the format defined under the relevant *Save* buttons and may be of a different resolution from the display. You can adjust the monitor display parameters and arrange for individual *Frames* to be saved to a specified location. You can create an animation file and save it to a specific location for playback through an external player. *RGB* and *Alpha Images* can be saved in 6, 8, or 24-bit depth and 8 or 24-bit respectively. You can set up the parameters required for outputting directly to a video recording system. Each button and data field found on the *Record Editor* are described below.

Data Overlay

This button allows you to overlay the rendered image with a word or phrase (20 characters maximum) which you can use for identification purposes. The *Frame* number of the image will also be recorded. Clicking the Data Overlay button activates the *Data Overlay Label* window.

Data Overlay Label Type an appropriate word or phrase in the window using the keyboard. The *Frame Number* will be automatically assigned to the label. If you do not enter any text, only the *Frame Number* will be displayed and recorded with the file.

Render Display

This button is a scroll bar which presents the available options for the displaying the rendered image. There are five options.

Render Display / None There will be no output to the display device.

Render Display / Toaster Display output must be directed to a Video Toaster system.

Render Display / 6-Bit HAM Display output is in 6-Bit HAM format.

Render Display / 8-Bit HAM Display output in in 8-Bit HAM format.

Render Display / Picasso II Display output is controlled by any Picasso graphics board. (Note: Picasso II was current when v3.5 was released).

Save ANIM File

Click this button to *Save* the *Scene* render as an *Anim* type animation file. The *Save ANIM File r*equester pops up in which you should enter the path and location of your animation file directory and the name under which you wish to *Save* the *Anim File*. LightWave will default to a directory called *Framestore*. This is a remnant of the Video Toaster system and a message will inform you that the directory can't be found. If you wish, you could create this yourself and establish the relevant path, otherwise, enter the path and name of your selected location.

ANIM File Name:

This window will display the path and name of the *Anim* file you intend to *Save*.

ANIM Type

This button is a scroll bar with six options, four of which require access to the Video Toaster system. On the assumption you will not have this equipment, the two *Anim Types* of interest are:

ANIM Type / 6-Bit HAM ANIM The animation file *Saved* will be *6-Bit HAM* format

ANIM Type / 8-Bit HAM ANIM The animation file *Saved* will be *8-Bit HAM* format

Lock Palette

This button is only active when the Video Toaster system is used. Animations can often be played at a higher speed if the *Palette* colours are *Locked*. That is, all the *Frames* have the same palette. This is usually acceptable for short animations or when there is relatively little colour change throughout the playback.

Lock Palette Frame

This button is only active when the Video Toaster system is used. It is used to select the *Frame* number whose palette will be used throughout the animation.

Loop ANIM

This button is only active when the Video Toaster system is used. It allows extra *Frames* to be rendered at the end of the series, which are delta compressed with the first *Frame*. This allows the animation to loop without disintegration of the *Frames*.

Begin Loop at Frame

This button is only active when the Video Toaster system is used. It sets the point of the loop.

Save RGB Images

Click this button to *Save RGB Images* of the individual *Frames* in the *Scene (Animation*). This is a useful way of saving all the *Scene* data so that they can be post-processed, converted into different image file formats or

compiled into different animation file formats externally of LightWave. Clicking the button pops up the *Save RGB Images Prefix* file requester. This requester will default to the Video Toaster's *Framestore* directory. You should therefore insert the path to the directory in which you wish to save the *Images*. Each *Image Saved* will be given the same *Prefix* name, followed by a three digit number corresponding to the *Frame* count. The first *Frame* rendered will be suffixed '000' whether or not *Frame* 0 is the first in the *Scene* to be rendered. Suffixes are added numerically according to the *First Frame*, *Last Frame* and *Frame Step* settings made under the Layout/*Render* button (see above). If *Image Saving* is on, the path to the *Image* directory is saved with the *Scene* file when you *Save* the *Scene*, or when you *Clear* the *Scene*, or when you *Quit* Layout.

RGB Image Prefix:

This window will display the path and file name *Prefix* of the *Images* to be rendered and *Saved*.

RGB Image Format

This button is a scroll bar which provides five options for the format of the *RGB* Images to be *Saved*.
The options are:

24-Bit IFF
24-Bit Raw
24-Bit Targa
6-Bit HAM
8-Bit HAM

Save Alpha Images

Click on this button to *Save* the *Alpha Images* (greyscale) of the individual *Frames* of the *Scene (Animation*). This provides the required data for compositing other *Images* with the images from the *Scene (Animation*). The *Save Alpha Images Prefix* file requester will pop up. This requester will default to the Video Toaster's *Framestore* directory. You should therefore insert the path to the directory in which you wish to save the *Alpha Images*. Each *Alpha Image Saved* will be given the same *Prefix* name, followed by a three digit number corresponding to the *Frame* count. The first *Frame* rendered will be suffixed '000' whether or not *Frame* 0 is the first in the *Scene* to be rendered. Suffixes are added numerically according to the *First Frame*, *Last Frame* and *Frame Step* settings made under the Layout/*Render* button (see above). If *Alpha Image Saving* is on, the path to the *Alpha Image* directory is saved with the *Scene* file when you *Save* the *Scene*, or when you *Clear* the *Scene*, or when you *Quit* Layout.

Alpha Image Prefix:

This window will display the path and file name *Prefix* of the *Images* to be rendered and *Saved*.

Alpha Image Format

This button is a scroll bar which contains two options for the format of the *Alpha Images* to be *Saved*.
The options are:

8-Bit IFF
24-Bit IFF

Fader Alpha Mode

Click this button to use the *Alpha* channel image to control a video fader. You will be warned of this when the button is clicked.

Save Framestores

This button is only active when the Video Toaster system is used.

Play Framestores

This button is only active when the Video Toaster system is used.

Framestore Comment:

This window is only active when the Video Toaster system is used.

Last Frame Displayed:

This window is only active when the Video Toaster system is used.

Serial Recording

This button is only active when the Video Toaster system is used. It instructs the Toaster to send a 'record' signal (via the Serial port) to a single-frame recorder. Each *Frame* is then recorded to tape as it is rendered.

Starting Position

This field sets the *Starting Frame* number for the Serial Recording routine (see above).

Record Setup 1

This button allows you to send a one-time via the Serial port to a single frame recorder (SFR) when it needs a single setup string command (see above).

Record Setup 2

This button allows you to send a one-time via the Serial port to an SFR when it needs two setup string commands (see above).

Record Command

This button specifies the string command sent to the SFR as each Frame is rendered (see above).

Frame Record Delay

This determines the number of seconds LightWave will wait after sending a record command before starting on the next Frame. This is to allow for pre-roll time on video tape recorders, typically 5 seconds. Video disc systems will not require this delay (see above).

Extra First Delay

This extends the Frame Record Delay for the first Frame of the Scene to allow for recorders which need more time for the first pre-roll (see above).

Continue

Click this button to return to Layout

The Options Menu

Clicking the *Options* menu button pops up the *Options Editor.* Here, you can adjust the resolution of the Layout screen. You can set the size of the Layout *Grid* by specifying the number of squares per side. The size of eachsquare can be varied and you can toggle the visibility of *Motion Paths* and the *Fog Radius*. If you wish to observe Layout redrawing its screen, you can slow this operation down enough to see it working. Each button and field on the panel is described below.

Layout Interface

The *Layout Interface* button is a scroll bar which provides six options for the resolution and colour of the Layout screen. The settings indicate the width and height of the screen in pixels, plus the number of available colours. Place the cursor on the bar and while holding down the LMB, place the pointer over the required setting. Release the LMB to implement that setting. The Layout screen will update accordingly. The available options are as follows:

672 x 432 (4 colors)
672 x 432 (8 colors)
800 x 600 (4 colors)
800 x 600 (8 colors)
1024 x 768 (4 colors)
1024 x 768 (8 colors)

The higher resolution settings will benefit from the use of a graphics board. The Picasso is supported by default.

Layout Grid

The Layout *Grid* is described in the main Layout chapter. You can control the visibility and size of this *Grid* by clicking on this button. The *Size* is expressed as the number of squares in the length and width of the *Grid*.

The button is a scroll bar with nine options as follows:

Off The *Grid* visibility is *Off*.
2 x 2 (squares/side)
4 x 4
6 x 6
8 x 8
10 x 10
12 x 12
14 x 14
16 x 16

Grid Square Size

This field defines the *Size* of each square of the Layout *Grid*. The default is 1.0 unit of distance (metre). Insert the required *Size* using the keyboard. The Layout *Grid Size* will adjust interactively. Note that the *Size* of the squares has an effect on *Object* movement and rotation in Layout. Small squares mean small steps in the *Move* control. Large squares mean large steps. This means a downward change in the *Size* of the squares may be necessary to facilitate precision in the *Move* command. The automatic adjustment of the *Grid Size* on loading a 'large' *Object* can cause problems when adding small *Objects* into the same *Scene*. Manipulating the small *Object* on a *Grid Size* dictated by a large *Object* can prove tiresome. In such cases, you should revert to a smaller *Grid Size* using the *Options Editor*.

Layout Background

This set of buttons controls the visibility of *Background* elements in the Layout views.

Blank

This button turns off the visibility of any *Background Image* assigned via the *Effects Editor*.

BG Image

This button enables the Layout *Camera View* to display any *Background Image* set via the *Effects Editor.* The *Image* is displayed in a dithered 2-Bit format (black and grey).

Preview

This button enables you to display a *Preview* animation as a *Background Image Sequence*. The *Background Image* will update according to the *Scene Frame* selected. The value of this facility is seen when combining video footage with elements of a LightWave Scene. For example, you could load an *Image Sequence* into the *Background Image* channel (*Effects Editor*) and generate a wireframe *Preview* of the *Scene* without its *Objects*, but with *Background Image* switched on. This will create a wireframe *Preview* of the *Image Sequence*, which should be *Saved* (e.g. in the *3D:Previews* directory). Now load the *Preview* into Layout and add the *Objects* you wish to integrate into the video footage. You can now arrange their motions etc. to fit the video sequence exactly.

Show Motion Path

Click this button to enable the display the *Motion Paths* of selected *Objects*, *Bones* and *Lights* in the Layout views.

Show Safe Areas

This button is relevant to the production of video recordings for display on a television screen. *Safe Areas* are indicated by a pair of concentric lines which circumscribe the *Camera's* view. The inner border is the *Safe Area* for text. The outer border is the *Safe Area* for any action or movement. The borders have been optimised for different designs of television and ensure that text and action elements will remain visible when the animation is transferred to video tape and replayed via tv sets. The areas are for guidance only.

Show Fog Radius

With this option selected, you can see the *Maximum Fog Distance* (*Effects Editor*), but only in the three orthographic *Views*. The circle will define the farthest boundary of the *Fog*. *Object* visibility outside this circle depends on the *Maximum Fog Amount*. With a setting of 100%, no *Object* will be visible to the *Camera* beyond this line. To see the *Fog Radius*, the *Fog* must be activated in the *Effects Editor* and you should ensure that the magnitudes of the *Grid* and the *Fog* settings are comparable.

Show Redraw

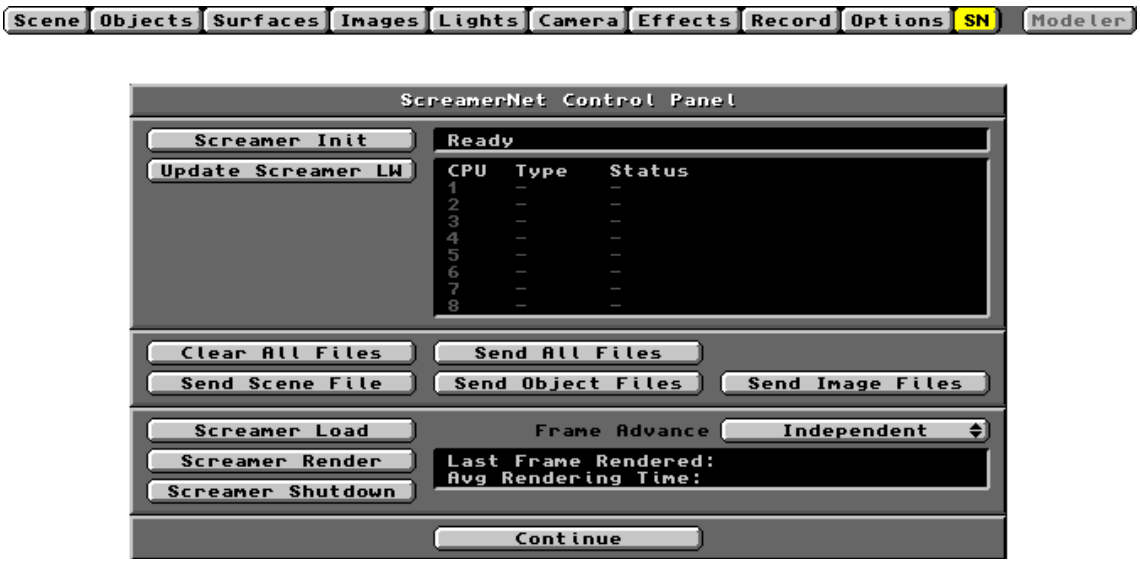
Click this button to enable the Layout screen *Redraw* to be displayed in real time. Normally, the Layout image is updated before it is displayed. This option enforces the display of the wireframe image as the computations are being made. Accordingly, the *Redraw* will be a little slower, but visible throughout.

Auto Key Adjust

This button instructs LightWave to automatically save changes in *Key Frame* data as you edit the *Scene*. This means that you do not need to click the *Save Key* button each time you have finished editing a *Key Frame.* Use this button with discretion, because a test setting will become the *Current* setting automatically.

Continue
Click here to return to Layout

The SN Menu

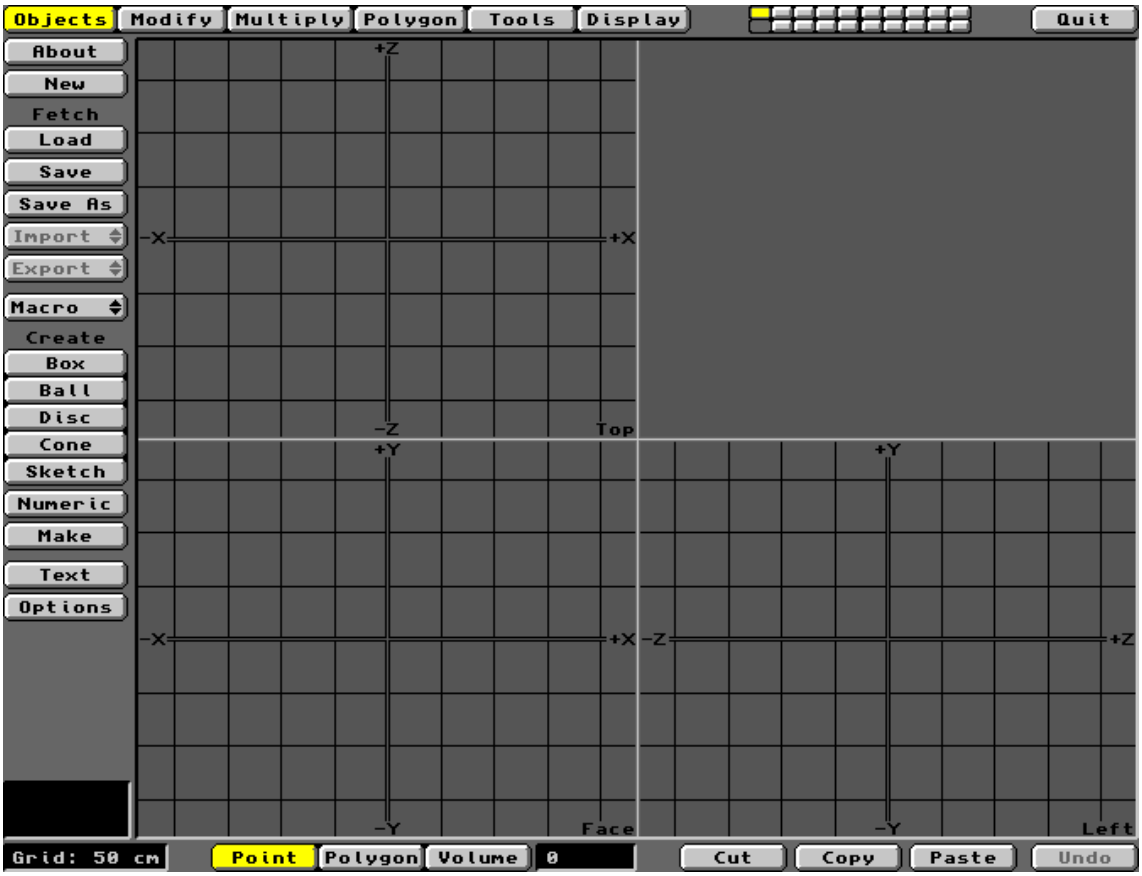


The *SN* button pops up the *Screamer Net* panel shown above. *ScreamerNet* is a software package needed for running LightWave renders on separate rendering engines such as the Raptor, etc. This software is not included with the LightWave package and will not be discussed further.

Warning

Clicking buttons on the *SN* panel may pop up requests for *Volumes* you don't have. If these requests fail to cancel, a reboot may be required to clear the problem.

The Modeler Interface



On entering *Modeler*, you are presented with a screen divided into four rectangles surrounded by several groups of buttons.

The three areas containing a dark grey grid are the *Edit Windows*. The plain grey area is the *Preview Window*. The centre of each *Edit Window* has a cross of double-lines labelled (+X, -X), (+Y, -Y) or (+Z, -Z). These are the *Axes* of three dimensional space (*X*, *Y* and *Z*), seen in two dimensions. In each window, the third axis is an imaginary line extending from the centre point, at right angles out of and into, the plane of the window. The centre point is the *Origin*, which has the coordinates (0,0,0), i.e.(X=0, Y=0, Z=0). Filling each window is the

Grid of squares used as a reference for *Object* scaling purposes. The notional size of each square is shown in the *Grid Size Window* in the bottom left corner of the screen (see *Options Menu* below).

The *Edit Windows* are labelled *Top*, *Face* and *Left*. These provide three orthographic views of the workspace, each oriented ninety degrees from the others. This system allows the position of any point in space to be located with great accuracy. Any *Object* loaded into *Modeler* will therefore be seen from these three views and its elements manipulated accordingly. By convention, the so-called *Main View* is that seen by the *Upper Left* window. The blank area at top right is called the *Preview Window.*

Top

The *Top* view (upper left) sees the *Object* from above, looking along the *Y* axis. Here, *X* is left/right and *Z* is up/down.

Face

The *Face* view (lower left) is a full frontal view, looking directly at the *Object*. By convention, this is the view along the *Z* axis of the workspace. Therefore, *X* is left and right and *Y* is up and down.

Left

The third view *Left* (lower right), places you to the left side of the *Object*, looking along the *X* axis. Thus, *Z* is left/right and *Y* is up/down.

Preview Window

The *Preview Window,* is presently blank because it is turned off. However, this can provide a 3D view of the *Object* being edited. It may be turned on using the *Options* button in the *Display* menu (see below). When it is off, it acts as a window re-size button. You can drag the workspace around by placing the cursor within the blank area and dragging it with the mouse (LMB). This is useful for enlarging a particular view to full screen. See notes below on the manipulation of the *Preview Window*.

The Button Banks

At the upper left of the Modeler interface is a horizontal row of six *Menu* buttons. These allow access to six different groups of *Command* buttons, which appear vertically on the left side of the screen. There are dozens of commands available here to allow you to manipulate files, create primitive 3D objects (*Ball*, *Box*, *Disc* and *Cone*), draw freehand shapes and invoke text manipulation. There are other commands which act on either *Points*, *Polygons* or both. How many of the command buttons respond is determined by the *Point, Polygon* and *Volume* mode selection buttons at bottom left. The manipulation of elements which combine to made a 3D object is done by Modeler using the 'Select and Modify' principle. Either the complete object, a group of polygons, a group of points, an individual polygon or an individual point can be selected and acted on by the many tools available in the Modeler interface. (See *Selected or Not Selected?* below) The *Menus* are described in more detail later.

The Layer Buttons

To the right of the *Menu* buttons is a double row of small buttons. These are the ten *Layer Buttons*. Modeler's enable objects to be placed in either the foreground (upper row) or the background (lower row) of the three *Edit Windows*. The *Layer Buttons* allow you to work on multiple screens as you edit *Objects*. Think of these layers as transparencies, each one stacked on top of another. You can choose to work in one layer, or several, with or without the visibility of the others.

The upper row contains ten *Front* layer buttons. In the lower row are nine *Back* layer buttons. The layers themselves are not numbered, but are considered to be layers 1 to 10 (left to right). At least one *Front* layer is active at all times (whether or not it contains an item). The rest may be active or not and can be placed in either location.

When a *Layer Button* is tagged with a *black dot*, it contains an *Object* of some kind. This *Object* may be a complex matrix of *Polygons,* or it could simply be a single *Point*. Select any *Front* layer (upper) button to activate that layer and its contents for editing. Select any other *Back* (lower) button and the *Object* it contains will be placed beneath the selected *Front* layer. Items placed in the *Back* layer cannot be edited and are used as a reference.

If you hold down the *Shift* key when pressing *Front* or *Back* buttons sequentially, each will remain *selected* (yellow). This allows you to manipulate several *Front* layers concurrently, or to see the contents of several *Back* layers concurrently. The contents of each or any layer can be saved separately under a unique *Object* name by selecting one layer at a time as a *Front* layer and *Saving* the contents to the *3D:Objects* directory.

In short, a layer may be in one of three states:

In Front - visible and active (it can be edited with the tools).....Upper row and Yellow

At Back - visible but inactive (it cannot be edited).....Lower row and Yellow
Off - invisible and inactive.....Either row and Grey

Layers can contain any number of *Points*, *Polygons*, *Curves* or *Objects*.

The contents of any layer can be simply *Cut* (and discarded), or *Cut* and *Pasted* into another layer, or *Copied* and *Pasted* into another layer using the three buttons at lower right. Each of these buttons has a single action *Undo* function. The *Undo* button also works with most of the *Point* and *Polygon* manipulation tools.

Selected or Not Selected?

It is important to understand how *Modeler* sees selected and unselected items.

Any part of a mesh highlighted (yellow) is selected and therefore active, whilst the rest is inactive. However, if no part of the mesh is highlighted, then LightWave considers it is all selected and active.

Selection and *Deselection* are two important parts of 3D modelling in LightWave. These states toggle, which means that the items you work with may be switched between the two. When you click on an unselected item with the left mouse button, you select it and it becomes active. When you click on a selected item, you deselect it and it becomes inactive. Active items will be affected by *Modeler*'s editing tools, while inactive items nearby remain unaffected. Active items are always highlighted in yellow. Inactive items are grey. This principle applies to the basic elements of the 3D *Object* (*Points* and *Polygons*) and to elements of the *Layout* and *Modeler* interfaces (mainly buttons).

The Selection Mode Buttons

The three *Selection Buttons* at lower left are key to *Modeler's* 'select and do' style of operation. They control the elements of an *Object* which will be acted upon by the various tools and operators.

Selecting *Point* mode limits your influence to *Points* only (singly or in groups).
Selecting the *Polygon* button limits your influence to *Polygons* only (singly or in groups).
The third button (*Volume*) toggles between *Include* and *Exclude*. When *Volume* is active the cursor turns into a bounding box, which can be placed around all or part of the *Object* in three dimensions using the mouse. Any operation will then be limited to the parts inside or outside the bounding box according to the toggle setting.

Select any element (point or polygon) of the an object by clicking on it with the left mouse button. When selecting a polygon with the cursor, all polygons connected to the point of the cursor will be activated. Judicious positioning of the cursor will ensure the desired result. Any 'extraneous' polygons can be deselected by second clicking on an edge not associated with the target polygon.

When selecting a point with the mouse, place the cursor reasonably close to the target. *Modeler* has a tolerance within which the target will be selected. If two or more points exist within the tolerance distance from each other, they may all activate. Deselect 'extraneous' points by re-clicking on them with the very edge of the cursor.

A 'global' selection method for both points and polygons is the *Lasso*.
In either *Point* or *Polygon* mode, holding down the *Right* mouse button will allow you to draw a rough circle or lasso around all the targeted items. Releasing the left button will cause all lassoed items to become active.
Active items can also be deselected using the Lasso method.
The *Lasso* can be used to define a *Volume* (see above). The 2D *Lasso* will actually extend along the axis of the edit window in which you draw it, enveloping a 3D space.

Holding down the *Shift* key will allow additional points or polygons to be added to already selected groups.

To select all points or polygons which are connected to the currently selected one, press the ']' key. The same result is obtained by pressing the *SelConn* (Select Connected) button in the *Display* menu.

Associated groups of *Points* or *Polygons* may also be selected or deselected via the *Stats* (Statistics) button located in the *Display* menu. Pressing *Stats* provides an information panel containing more selection buttons (+ select) (-- deselect). These allow you to select and deselect items based on point count, polygon count, surface name grouping, volume inclusion or exclusion. It can also be used to find non-planar polygons.

To locate/select /deselect non-planar *Polygons*, press *Stats* in *Polygon* mode. The non-planar listing is given at the bottom of the panel. To select any non-polar polygons, press the (+) button. The panel will disappear and the polygons will be highlighted. By default, *Modeler* considers any polygons more than 2% out of planar alignment are regarded as non-planar. This parameter is determined by the *Flatness Limit* value. This value can be adjusted in the *Options* command under the *Objects* menu.

At bottom left of the *Modeler* interface are two black mini-screens. The upper one displays the *X*, *Y* and *Z Coordinates* of the cursor and the *Status* of the operation in progress. Since each window shows a two-dimensional view, only two coordinates at a time are given, depending on which window the cursor is within. When you engage a *Modeler* tool such as *Rotate*, the coordinates window will display the amount of rotation applied in degrees. Other tools may display the angle of bending or twisting applied to the *Object*, or the distance the cursor has moved, or the size/stretch factor.

The lower black mini-screen is the *Grid Size* box. The *Grid Size* box changes its display automatically, so you always have a useful reference size on screen. For example, with the *Grid* sized at 1 metre increments, you may wish to edit an *Object* only a few centimetres big. This *Grid* is obviously too large a reference for accurate work. However, all you need to do is zoom into the *Object* and the *Grid Size* will automatically re-orientate itself to a smaller setting. To change the *Grid Size*, zoom *In* or *Out* via the *Display* menu. When the *Grid Size* shows a 10 millimetre setting, you can work on the *Object* knowing the exact size of any additions by reference to the *Grid*.

The *Blank Area* located below the vertical *Command Button* set serves the function of a *Deselect* area, toggling *Off* the selected status of *Points*, *Polygons* or *Volume*s that have been selected earlier. Just click inside the area with the mouse. This gadget allows for mass deselection without having to laboriously deselect *Points* or *Polygons* on a more or less individual basis. Note that any blank area of the *Modeler* interface works in this way. Certain command functions must be closed (deselected) before structural deselection will work. In these cases, click on the command button again, then in any *Blank Area*. You can also click with the *Right* mouse button on either the *Points* or *Polygons* mode button to deselect them.

The Preview Window

Activate the *Preview Window* by pressing the *Options* button, found under the *Display* menu. A panel pops up in which various options for using the window can be selected. It can then be used as a fourth edit window for item selection, or as a preview screen showing a *Static* or *Moving* (oscillating) image of the edit *Object*. This image can be in simple wireframe or a solid representation of the *Object*. Further, when activated, the window displays a *Black Circle* surrounding the image. This is *Modeler's* 'virtual trackball', which enables you to rotate the *Object* without affecting it in the *Edit Windows* proper. With the mouse, drag the circle from top to bottom, left to right, or around the perimeter to rotate the image the three axes. Each adjustment of the image requires significant computation and whilst this is going on, the image will disappear. When the new image is computed, it will re-appear automatically. An oscillating image is often useful in checking the relationships between parts of the *Object* and how it will appear in its rendered state. Further computation and therefore time is needed to render several wireframe or greyscale images for this 'mini -animation'.

The Objects Menu

Clicking the *Objects* button brings three groups of buttons into the command bank. Most are concerned with *Object* creation and file handling, but the very top pair are general use buttons.

About

About provides a little information about the *Modeler* software.

New

New will clear all data from the *Modeler* interface in readiness for a new job. The *Edit Windows* are reset to their default configuration.

The rest of the buttons are grouped under two broad headings: The *Fetch* group and the *Create* group. Group headings like these appear throughout the command banks and provide a general idea of what the group will do. For example, the *Fetch* group are commands relating to *File* access (between Modeler and the hard drive and between Modeler and Layout). Here you will see familiar file loading and saving commands.

The Fetch Group

Load, ***Save***, ***Save As***

Clicking on any of these buttons will pop up a file requester for you to select or enter filenames. The default path for items you will handle here is *Toaster:3D/Objects* and will appear automatically in the requester. If you wish to load or save to a different location, you must type in the path to the required directory using the keyboard. Note that Modeler will not accept foreign file types directly. However, Layout has built-in converters for six externally generated file types. If you use Layout to load a foreign *Object*, this can then be imported into Modeler for manipulation and saving as a LightWave *Object*.

Import and ***Export***

These buttons enable you to communicate with the Layout interface using Modeler. Layout must be running and in order to *Import* any object into Modeler, it must already be loaded into Layout. *Import* and *Export* are unique to the Amiga's multitasking capability. Other platforms require you to *Save* all objects to disk before they can be *Load*ed into either interface.

Macro

Macro provides access to *Modeler's* list of built-in command scripts as well as the ability to load in custom Arexx scripts. These scripts contain sequences of commands that have been grouped together into a single, short-cut operation.

The Create Group

The *Create* group of commands are related to *Object* creation. For example, generation of object *Primitives* - *Box*, *Ball*, *Disc* and *Cone* can be achieved automatically from the relevant button and the numeric panel which pops up. It also provides controls for 3D *Text* manipulation and to *Options.* Here, also you will find the *Sketch* button to allow freehand sketching of shapes, and the *Numeric* button which facilitates input of numeric data. *Numeric* data is used to modify the effects of other commands and tools in the *Object* menu. The *Make* button is used to complete the computation of polygons following the manipulation of the commands and tools.

Box, Ball, Disc* and *Cone

The 'primitives' buttons (Box, *Ball*, *Disc* and *Cone*) provide access built-in shapes for use as starting points for more complex *Object* creation. By selecting the appropriate button, *Modeler* will automatically construct that shape in three dimensions according to the parameters you set, either by the mouse cursor or by *Numeric* input. An example is described later (see *Simple Objects*).

Sketch

Sketch allows you to draw freehand shapes, which *Modeler* converts into *Curves* or *Polygons*. Place the cursor in any window and drag with either mouse button to draw the shape. After selecting *Sketch*, you can choose between *Curve* and *Polygon* creation via the *Numeric* button. If you draw with the *Left* mouse button, you will get a dotted line and you must click *Make* to complete the job. If you use the *Right* button, a solid line will be drawn, which will automatically be completed by *Modeler* as a *Curve* or *Polygon* according to the *Numeric* setting. The value inserted in the *Plane* box controls the depth that the line will be drawn *below* or *above* the plane of the window used for the *Sketch*. This positioning will only be apparent in the two complementary views. The number of *Points* associated with a sketched line will vary with the speed at which it is drawn.

Numeric

Numeric is the button to click when you have already opened another command which needs numeric parameters or other data to be entered by you. This will allow you to create objects with very specific parameters in terms of Polygon size, location etc.

Options

Options deals with polygon *Types*, *Surface* names, the *Flatness Limit* and *Curve Division*.

Text

Text gives access to *Adobe/Postscript Type 1* fonts directly from *Modeler*, converting letters, numbers, symbols and words into flat 3D *Objects*. Pressing *Text* pops up a dialogue box in which you may type a word or phrase. This will be drawn by LightWave using the *Font* indicated in the scroll bar. Additional fonts can be selected by scrolling down the list which appears when the scroll bar is selected. The list may be extensive and continue beyond the visible panel. If so, an *up* or *down* arrow will indicate that further names are available. *Fonts* can be loaded and removed via this dialogue box. The *Fonts* directory can be entered into the file requester which pops up after pressing *Load*. *LightWave's* default fonts directory (*ToasterFonts/SoftMaker*) is listed automatically.

The current *Font* can be deleted from the options by pressing the *Remove* button. The *Corners* function allows improved bevelling of certain types of *Font*. Some Postscript fonts are given additional points near their corners when converted into 3D objects. This leads to rendering inaccuracies. Use the default *Sharp* for most situations. The *Buffered* button retains the extra points, should you require them.

The Modify Menu

Pressing the *Modify* menu button changes the command buttons on the left into a three groups of new buttons which are concerned with modifying *Objects* on screen. As you move downwards from the *Position* group, through the *Flex* group to the *Deform* group, the tools become more and more specific to defined areas of the *Object*. The upper group concerns *Objects* as a whole, the middle group acts on only a portion of the *Object*, while the lower group acts only on portions which you pre-define.

The Position Group

Move

Move will move the selected *Object* to a different location on the screen. Select the item to move, click *Move* and place the cursor in an *Edit Window* and drag with the mouse. The *Coordinates Window* displays the distance moved by the *Object*. Holding down the Ctrl key limits the move to up/down, left/right or diagonally.
When moving *Points* (press *Points* mode button first), *Move* is complemented with an additional tool called *Jump*. *Jump* (j) is a keyboard only command and moves a selected *Point* to the cursor instantly. If several *Points* are selected, they will all *Jump*, the last selected *Point* jumping to the cursor, the rest retaining their positional relationship with each other. Like many commands, *Move* has a *Numeric* parameters function. Press *Numeric* to pop up the dialogue box.

Rotate

Rotate causes the selected item to rotate about a single axis. Select the item, click *Rotate* and place the cursor in an *Edit Window*. Drag the mouse to rotate the item. The degree of rotation is given in the *Coordinates Window.* Holding down Ctrl limits rotation to 15 degree jumps. The axis of rotation is the cursor. *Rotate* also has a *Numeric* parameters function (press *Numeric*).

Size

Size changes the dimensions of the selected item equally on all three axes. Select the item, press *Size* and drag the mouse to resize the item. The size factor is given in the *Coordinates Window*. If the selected item is away from the *Origin* of the *Edit Window* used, *Size* will cause the item to move in the direction you drag. Using *Numeric* will allow you to resize by a precise factor and to specify the final coordinates of the resized item. Do not confuse *Size* here with the *Size* button in *Layout*. *Modeler* affects the *Object's* actual size, which is saved with the *Object*. Any resized *Object* in *Layout* is saved as part of the *Scene* script file and makes no change to the actual *Object*.

Stretch

Stretch will stretch an *Object* , a *Polygon* or group of *Polygons* by dragging the mouse in an *Edit Window.* Holding down the *Ctrl* key modifies *Stretch* to one axis. If the mouse is moved along one *Axis*, stretching occurs along that Axis only. Unlike *Size*, *Stretch* does not affect all three axes equally. It stretches equally along the two *Axes* of the *Edit Window* used. *Objects* near the cursor stretch less than *Objects* a greater distance from the cursor.

Drag

Drag is a *Point* specific tool, which can be use to move *Points* individually, when nothing is selected. When several *Points* are selected, only those may be dragged. If a *Polygon* is selected, only those *Points* attached to that *Polygon* will be dragged.

The Flex Group

Shear

Shear causes *Polygons* or *Objects* to tilt. An example is the conversion of standard Roman Text into italic text. Both Ctrl and *Numeric* are modifiers for *Shear*. There are several modifications available via *Numeric*, which controls the *Axis* of shear, how much of the object will be affected (*Range*), whether one end of an object or the other will be affected (*Sense*) and whether the shear will have curvature at the fixed end or the sheared end (*Ease*-factor).

Twist

Twist allows you to twist an object as if you grabbed it at one end and pulled it in a circular motion. The object twists around the point located by the cursor as you click on the left mouse button. *Ctrl* constrains twisting to 15 degree jumps. *Numeric* provides parameters analogous to those found in *Shear*.

Taper 1

Taper 1 (uniform taper) will taper an object or polygon equally on two axes, something like a pyramid or cone. The axes of *Taper 1* are those defined by the *Edit Window* used. It also has *Numeric* parameters analogous to *Shear*.

Taper 2

Taper 2 (disjointed taper) will taper an object or polygon on any two axes independently.

Bend

Bend will bend an object as if you took it in two hands and bent it. It works also on polygons, though any with four or more edges may create non-planar surfaces, which induce rendering errors. The axis *perpendicular to* the *Edit Window* is the axis around which bending takes place. There are *Ctrl* and *Numeric* modifiers to the *Bend* command.

The Deform Group

Magnet

Magnet can push or pull points or polygons towards or away from an *Object* as if influenced by a magnetic field. Click *Magnet*, move the cursor into an *Edit Window* and drag out a *Bounding Box* with the LMB. This defines a magnetic column that extends into and out of the plane of the *Edit Window.* Move the cursor into another window if you wish the magnetic influence to be limited within a defined three dimensional box. With the box in place around part or all of the object, drag the mouse with the right mouse button. *Ctrl* causes the magnetism to concentrate more closely to three axes: the two visible in the Edit Window and a diagonal line across the window.

Vortex

Vortex is a special kind of rotation tool, which spins the contents of a bounded area, more in the centre and less towards the outer edges. Vortex uses both the LMB and RMB. With the LMB, you define the *Bounding Box*. You manipulate the vortex with the RMB. *Ctrl* constrains the change to 15 degree steps. There are other *Numeric* parameters.

Pole 1

Pole 1 is a special sizing tool that effects a bounded area. It acts on two axes equally and pushes or pulls the contents of the bounded area in unusual ways, somewhat like a pulsing magnet. It uses both the LMB and RMB. The LMB defines the *Bounding Box*, while the RMB performs the deformation. If the *Bounding Box* is

centred exactly on the *Object* and the cursor is centred with the *Object*, *Pole 1* will resize the *Object* just as if you had used *Size.* However, once the cursor is placed off-centre, *Pole 1* acts as expected.

Pole 2

Pole 2 acts like *Pole 1* but on two axes independently. It is useful for producing more organic shapes.

The Multiply Menu

Multiply contains commands which can produce multiple copies of a 3D shape. Certain of the tools in this menu make use of movable *Edit Axis* around which a given operation will be performed. After you have selected the item and chosen the command, you have the option of performing the multiplication manually (using the mouse), or numerically (via *Numeric*).

Extrude

Extrude will extend the selected polygon(s) along any axis, providing depth or thickness. A typical use is to convert 2D text polygons into 3D text objects. Select the item to manipulate. Select *Extrude* and move the cursor into an *Edit Window* and left-click. A light grey (+) sign appears under the cursor and a broken-line *Bounding Box* surrounds the object. In the other two windows, the *Bounding Box* will indicate the current extent of the latent extrusion. You'll also see a T-shaped bar in these windows. Click directly on the 'T' and drag it to increase or decrease the extrusion distance. When this is set, select *Make* (or press the RMB). The extrusion will be completed. *Extrude* has a *Numeric* function which allows you to set the extent of the extrusion, the number of segments into which the extrusion will be divided and the axis along which it should take place. Closing the dialogue box with OK will produce a broken-line *Bounding Box.* If the latent extrusion distance is acceptable, press *Make*. If you extrude polygon *Curves*, the *Curve* will be deleted after the operation. If you need the *Curve* again, *Save* it before extrusion.

Lathe

Lathe will spin an item around an axis, creating duplicate cross-sections in the process. These are connected together to provide the outer surface. *Lathe* may be conducted using the mouse or via the *Numeric* dialogue box. Select the item to be spun, select *Lathe* and move the cursor into an Edit Window and left-click. A light grey (+) sign appears under the cursor, indicating the axis around which the latheing will take place. In the other windows, a light grey bar appears, extending across the window. These are the latheing axis from the other two views. Select *Make* to complete the latheing process. The *Numeric* option of *Lathe* provides a dialogue box into which you enter various parameters. *Lathe*ing will normally provide a 360 degree circular extrusion of the selected item. If you wish to reduce or increase this rotation, you can enter a suitable values in the *Start Angle* and *End Angle* to control the extent of the latheing process. *Sides* is the required number of segments that will complete the latheing process. It's like the slices in an apple pie. An *Offset* value will shift the original shape by this quantity as it is lathed. *Offset*s always operate along the latheing axis. Using *Offset* gives results resembling a coiled spring.

Mirror

Mirror will make a mirror-image of the selected item. This is useful for creating objects which have a plane of symmetry. Select the item to manipulate, select *Mirror*, move the cursor into an *Edit Window* and left-click. The is activates the edit axis. This is a dividing line which follows the cursor and extends through two windows. It will alternate between a horizontal and vertical position as it nears the edges of the *Edit Windows*. This allows you the obtain the correct orientation for the *Mirror* process. Position the axis where required and select the *Make* button (or click RMB). Analogous manipulation may be achieved via the *Numeric* option of *Mirror*.

Make

This button is used in conjunction with others in this group (see above).

The Sweep Group

Bevel

The *Bevel* command adds a bevelled edge to selected polygons. It shifts the edges of selected polygons by the *Shift* value and moves them forwards/backwards by the *Inset* value. It then connects the new edges with more polygons. The *Shift* is made in the direction of the polygon's *Normal*. A *Bevel* with an *Inset* value of 0 is the same as an *Extrusion* equal to the *Shift* value. Select the one or more polygons or the entire object, press *Bevel* and enter values into the dialogue box. The *Shift* and *Inset* values may be positive or negative. Positive

Inset causes the edges to move towards the centre of the polygon. Negative *Inset* moves the edges away from the centre. Positive *Shift* moves the resultant polygon in the direction of its *Normal*. Negative *Shift* does the opposite. In all cases, the outer edge of the bevel is defined by the location of the parent polygon.

The use of positive values to *Bevel* text reduces its facial dimensions and letters with a narrow face may not respond too well. Here, the bevelled edges may tend to cut into each other, making for a clumsy appearance. A negative *Inset* retains the facial dimensions of the text and the bevels go outwards, but this can also cause problems with some letters. Either way, use values which are a relatively small proportion of the text's size.

Bevelled text is usually constructed using a combination of the *Bevel* and *Extrude* tools. However, the results you obtain may not what you expect. This is due to the relocation of the face polygons following the *Bevel* operation. Unless you are absolutely sure which polygons are supposed to be in the face of the text, the *Extrude* result can be very unstisfactory. The sides of the text may appear fragmented or grooved. This is not an error, it is simply the result of extruding polygons which have been relocated via the *Bevel* command.

To bevel *Text* reliably, please refer to *Tutorial 11*.

If the *Inset* value is too large, the bevels will start to cross each other in the front of the template polygon. Try using a smaller bevel. *Bevel* works on both single- and double-sided polygons. In the latter case, it will bevel on both sides in the facing direction of each polygon. However, double-sided polygons in closed 3D objects (e.g. sphere) will not be affected. *Bevel* can also create additional detail and surfaces. Try *Bevel* on a standard sphere for example, to get some interesting results. The *Metaform Tutorial* has further ideas on the use of *Bevel*.

Sm Shift

Sm Shift (*Smooth Shift*) adds a specific type of detail to a polygon object or mesh. It creates duplicates of the selected polygons(s) a certain distance from the original(s). The copies replace the originals and the duplicates will be shifted along an axis which corresponds to the *Normal* from the original polygon(s). Select some polygons, press *Sm Shift* and enter values in the pop up box. Enter an *Offset* value (positive or negative) and if desired, change the *Maximum Smoothing Angle*. Click *OK*. The *Maximum Smoothing Angle* (*MSA*) determines whether adjacent polygons are smoothed out or left with a sharp cut-off.

Maximum Smoothing Angle (*MSA*)

The *MSA* defines the maximum angle that can be smoothed across by tools like *Smooth Shift*. You can visualise the *MSA* as the angle between any two adjacent *Normals* (extended until they intersect) which the Modeler considers before applying any *Smoothing* action. Two polygons whose *Normals* are inclined to one another at an angle *greater* than the *MSA* retain a sharp edge, rather than the join being smoothed over. The *MSA* default setting is 89.5°. This means that any polygons inclined at 90° to each other will remain at right angles after smoothing is applied.

For a new approach to implementing the MSA, refer to details on the ***Smoothing Threshold***©, discussed In Tutorial 3.

Path Ext

Path Ext (*Path Extrude*) will extrude and skin a polygon or object along a *Motion Path* created and saved in *Layout*. This is different from *Path Clone* described below. Select an item and then select a *Motion* (*3D/Motions* directory). A second menu appears in which you enter the desired first and last frames of path and a frame step. Then select OK. The item will be extruded along the specified path. Polygons must be single-sided to work with this tool.

Rail Ext

Rail Ext (*Rail Extrude*) extrudes a 2D or 3D object along a *Curve* path. This is a *Single Rail Extrude*. It can also extrude the object along multiple *Curves* (*Multiple Rail Extrude*). Arrange the object in a *Front Layer* and a *Curve* (path) in a *Back Layer*. Click *Rail Ext* and select options in the pop up box. Then click OK.
To work properly, this tool requires the following conditions:

The *Object* to be extruded is in the *Front Layer*
The extrusion path (polygon *Curve*) is in a *Back Layer*.
The *Object* should be orientated so that it is at the starting point of the extrusion path.

Segments is the number of slices in the extrusion. *Automatic* determines an optimal number. *Uniform Lengths* distributes the specified number of *Segments* evenly along the extrusion. *Uniform Knots* distributes the specified number of *Segments* evenly between the knots which determine the shape of the *Curve*. *Oriented on* causes all segments of the extrusion to angle themselves automatically, so that they are aligned with the *Curve* as it turns. *Oriented off* causes all segments to remain in the same orientation as the template so that they remain faced in exactly the same way.

Patch

Patch creates a smooth surface mesh connecting three or four *Curves*. It is a powerful command which will give an endless supply of free-form meshes for later manipulation. The *Curves* used need not be planar in 3D space, though they must form at least one closed area as seen in the *Edit Windows.* Whilst they do not have to be joined end to end, they do need to be joined somewhere along their length. At the points where the *Curves* overlap, those points must be *Merged*, or *Welded*. For example, a pair of S-shaped *Curves* placed some distance apart in an *Edit Window* could have their ends joined together using two straight *Curves*. Their eight overlapping *Points* should then be *Merged* (or *Welded* in overlapping pairs) into four *Points.* The *Patch* command will create a billowing curtain-like mesh. It will however, be planar. To create three-dimensional meshes, the *Curves* must be manipulated in all three *Edit Windows.* This can take a while to master and is best done by making *Points* into *Curves*. See *Create/Points* (*Polygon* menu) and *Curves/Make* (*Tools* menu).

To invoke the *Patch* command, all three/four *Curves* must be formally selected (i.e. highlighted yellow). Note that this differs from normal selection conditions. Use any *Polygon* selection method to do this. When you click the *Patch* button, the *Patch* data panel pops up. The *Numeric* options of *Patch* require you to input the following:

Perpendicular is the number of segments (divisions) that will be perpendicular to the last selected *Curve*.
Parallel is the number of segments that will lie parallel to the last selected *Curve*.

Because *Patch* is such a useful but complex operation, it is illustrated in more detail in Tutorial 8.

Skin

Skin allows you to cover a series of wireframe shapes with polygons. The shapes do not need to have the same number of points in common. *Skin* works best when you use single-sided polygons for the wireframes. Create or select a group of polygons (or object outlines) in the order that you want them to be connected. Select *Skin* to create an outer surface for the outlines. *Skin* does not require that the number of segments it connects have the same number of points. There is no *Numeric* option for this tool.

Morph

Morph creates a number of connected, intermediate polygons between two polygons having the same number of points. The connection will have a skin as if you had used the *Skin* command on them. The polygons should be single-sided for optimal results. *Morph* requires numeric input. Enter the number of *Segments* (divisions) you wish to use for the morph. Click OK to complete the changes. Morphing with *Curves* treats them as straight-edged polygons and the morph will be linear.

The Replicate Group

Clone

Clone duplicates an item one or more times using specific increments of distance, rotation and scale (size). This allows you to a create spiral staircase out of a single step, for example. *Clone* requires *Numeric* data input to work. Enter data into to pop up panel. Enter the *Number of Clones* you require. Enter the distance (*Offset*) between each clone and the next along the X, Y, or Z axes. Enter the degree of *Rotation* between one clone and the next. Enter the amount of scaling for each copy in the *Scale* field. Enter the coordinates of the *Centre* point around which the *Clone* operation will take place. Immediately after a clone operation, you can *Lasso* the entire group and they will be selected in the order they were created. This means that you can *Skin* them if you desire. After skinning, the internal framework should be deleted to reduce memory demand.

Array

Array duplicates a selected item a number of times using specific increments of distance. This allows you to create a grid or matrix using copies of the object. *Array* requires *Numeric* data input to construct the matrix. Enter the number of duplicates to be made along each axis in the *Dimensions* field. *Offset* is the distance between each copy of the object. In *Automatic* mode, this distance is determined by the object's own dimensions. In *Manual* mode, you select the spacing between the copies.

Rail Cln

Rail Cln (*Rail Clone*) will clone a point, polygon or object along a curved path (single rail clone) or multiple paths (multiple rail clone). The item to be treated must be in the *Front* layer. The *Curve* path(s) should be in a *Back* layer(s). The item and the *Curve* should be oriented (in the appropriate view) such that the item is directly above the *Start* point of the *Curve* path. *Rail Cln* creates a number of unconnected copies of the item, evenly

spaced along the *Curve*(s). Both clone operation require *Numeric* data input. Enter the number of clones you want in the *Segments* fields.

For *Single Rail Clone*, there are three distribution options for the clones. *Automatic* will place a number of clones (based on the *Curve Subdivision Value* in the *Objects/Options* menu) along the *Curve* according to its knot spacing. *Uniform Lengths* will distribute the specified number of clones evenly along the length of the *Curve*. *Uniform Knots* places an equal number of clones between the knots of the *Curve*. The *Oriented on* option aligns the clones at right angles to the direction of the *Curve* at the point they occur. *Oriented off* aligns the clones in the same direction as the template item, so they all face the same way.

Multiple Rail Clone also requires you to specify the number of clones required. There are four distribution options. *Automatic* will place a number of clones (based on the *Curve Subdivision Value* in the *Objects/Options* menu) along the *Curves* according to their knot spacing. *Uniform Lengths* will distribute the specified number of clones evenly along the length of the *Curves*. *Uniform Knots* places an equal number of clones between the knots of the *Curves*. *Strength* determines how strongly the rail *Curves* vie for control of the point locations. The effect is very subtle. In most cases a value of 2 is appropriate. The *Oriented on* option aligns the clones at right angles to the direction of the *Curves* at the point they occur. *Oriented off* aligns the clones in the same direction as the template item, so they all face the same way. If the rails in the back layers spread apart along any axis, or axes, then the clones will automatically stretch along those axes as well. If you wish to have the clones scale equally, turn *Scaling on*.

The Polygon Menu

This menu contains a selection of tools to *Create*, *Revise* and *Transform* polygons in the following ways.

Create points in 3D space, which can then be converted into polygons (*Points*)
Insert or delete points from polygons (*Add Point*, *Remove Point*)
Assign surface names to polygon (*Surface*)
Attach or separate polygons (*Attach*, *Detach*)
Manipulate the surface normals of polygons (*Align*, *Unify*, *Flip*)
Split polygons into several smaller polygons (*Subdivide*, *Triple*)
Remove polygons without removing their points (*Remove*)

The Create Group

Points

Points creates *Points* within Modeler's 3D workspace. Note that this *Points* command button is different from the *Points* selection (mode) button at the bottom of the Interface. The *Points* mode button is activated when you press the *Polygon/Points* command, because you are working specifically with *Points*.

After pressing the *Points* command button, place the cursor into an *Edit Window* and move it to the desired location for the *Point*. Click the LMB to stamp a yellow crosshair. This cross-hair marks the position of the latent *Point* and may be discarded by left-clicking elsewhere in the window. Pressing the *Make* button generates the *Point*. If you place the cursor and click with the RMB, the *Point* is generated immediately.

After placing *Points* , they can be converted immediately into a *Polygon* by pressing 'p' on the keyboard. Otherwise, deactivate *Points* mode by pressing the *Points* command button again (remember the toggle principle for using command buttons) and press *Make*. A *Polygon* will be drawn, whose outline corresponds to the order in which the *Points* were created.

If you wish to create a *Polygon* from pre-existing *Points*, enter *Points* mode and select the relevant *Points* in the *Edit Window* using the cursor, holding down the LMB throughout. Alternatively, several *Points* can be click-selected by holding down the *Shift* key. Once all the relevant *Points* are selected, press the *Polygon/Make* button.

Polygons created from hand-placed *Points* are always single-sided, regardless of the setting of the *One Side* and *Two Sides* buttons in the *Objects/Options* menu. However, you do have influence on the direction the newly created *Polygons* face. To ensure the *Polygons* are rendered as visible polygons in *Layout*, select the *Points* in an anti-clockwise direction. Clockwise selection of *Points* gives backward-facing *Polygons*.

Make

Make completes the creation of *Points* and the creation of *Polygons* from *Points* (see the *Points* command above). When creating a *Polygon*, the *Points* must be selected first. In order to use the keyboard equivalent of *Make* (*Return*), you must first exit *Points* creation mode. Do this by re-selecting (toggling) the *Points* creation button first.

Remove

Remove allows you the delete a *Polygon* from the workspace without removing the *Points* at each vertex. Select the *Polygon*(s) you want to eliminate an press the *Remove* button. The *Polygons* are removed, leaving behind only their component *Points*.

The Revise Group

Add Pnt

Add Pnt (*Add Point*) Will add more *Points* to selected *Polygons*. The tool has two forms of operation: proximity and distant. If you wish to add *Points* in the immediate vicinity of a *Polygon*, simply select the *Polygon*(s) and use the *Add Pnt* command. The cursor will change to a arrow with the word 'To' on it. Simply place the arrowhead on the edge of the *Polygon* where you need a new *Point*. Click the LMB and a *Point* will be placed there. Continue adding more *Points* to this or any other selected *Polygon*.

If you wish to add a *Point* to a location some distance from a *Polygon*, first select it and then press the *Polygon/Points* creation button. Create a new *Point* at the desired location by following the instructions under the *Points* command above. Click *Add Pnt* and the cursor will change into a 'To' arrow. Click the arrowhead on the edge of the *Polygon* that needs the new *Point*.. Click with the LMB to incorporate the *Point* into that edge of the *Polygon*. If you wish to add several free-standing *Points* to a *Polygon*, you can select them (in an anti-clockwise order) and add them all in one step. *Points* may also be added to *Curves* in the manners described.

Rem Pnt

Rem Pnt (*Remove Point*) Will separate *Points* from the *Polygons* to which they belong. The *Points* are not deleted from the window (use *Cut* or the keyboard *Delete* command (z) to do this). In *Polygon* mode, select the *Polygon* and then change to *Points* selection mode. Select one or more *Points* you wish to remove and click the *Rem Pnt* button. The two *Points* either side of the removed *Points* will become joined. *Curves* can be treated in a similar way.

Attach

Attach will add a Polygon to the surface of another Polygon. Select the two Polygons you wish to attach. Click the Attach button and the cursor will change to a 'To' arrow. Click the arrowhead on the Polygon you want to be the 'attached' . The *Polygons* become attached to each other. This command performs a similar job to the 2D and 3D *Drill* operators described under the *Tools* menu, which will probably be preferred.

Detach

Detach is used to separate *Polygons* which were attached with the *Attach* command. Select the attached *Polygons* and click on the *Detach* button to separate them.

Split

Split divides a *Polygon* into two smaller polygons. In *Polygon* mode, select a *Polygon*. Change to *Point* mode and select two *Points* which do not share any edges. Click the *Split* button to split the *Polygon* along the line joining the points. A *Curve* can be *Split* by selecting it, then selecting the *Point* on the *Curve* at which you want to split it and click ing the *Split* button. The result is two *Curves*.

Merge

Merge will join two *Polygons* sharing a common edge. A common edge is one where two or more *Points* belong to two or more *Polygons*. Select one or more *Polygons* and click the *Merge* button. After merging *Polygons*, you should check that the resulting *Polygon* is planar, to avoid rendering problems. Select the *Polygon* and press the 'w' key to pop up the *Polygon/Statistics* panel. Here you can search for and highlight any non-planar *Polygons*. Also see the *Display/Stats* command about accessing to the *Statistics* panel. *Merge* can also be used on *Curves*. Select two adjacent *Curves* that share a common endpoint (they must be endpoints not inner points) and click *Merge*. They will merge into one continuous *Curve*.

If you attempt to *Merge Polygons* and get an error message, check that the *Polygons* do actually share a common edge. They may be overlapping in space, giving the appearance of a common edge. If this condition is met and the command still fails, then check that they are not simply duplicate points. Perform a *Point Merge* operation. Sometimes the points of two polygons overlap, making them look like a common point. *Merge* will operate on both coplanar and non-planar polygons. However, the resulting *Surface* may not render very well.

The Transform Group

Surface

Surface is used to define (or change) *Surface* names in a single operation. After selecting a group of *Polygons* you wish to define as a separate *Surface*, click the *Surface* button. A pop up panel allows you to enter a new name for the *Surface* (unnamed *Surfaces* are called 'Default'), or select and *Apply* an existing name from the scrolling list, or *Rename* a surface with a new name. Be careful to select only those *Polygons* you want to give a different name. If you omit to select a group of *Polygons*, then ALL *Polygons* are active and will be named accordingly. The pop up panel displays a list of all available surface names, including those you create using other surfacing controls. This is convenient for assigning names as you build up an object, because all new Polygons created after you exit the *Surface* function will have the last used name applied to them. An alternative approach is to design the complete object first and then select sections you wish to assign different surfaces to and name them one after the other. This method can be tedious with very complex objects, but it is a matter of personal preference.

Triple

Triple converts all selected *Polygon*(s) into triangular *Polygons*. For example, a rectangular or square *Polygon* will be split diagonally into two triangles. As noted elsewhere, triangular *Polygons* are planar and always render successfully . They are the preferred configuration for polygons undergoing deformation using the *Bones* function in *Layout* . Quadrangular (or higher order) polygons can be pushed out of planarity under the influence of *Bones*, which makes rendering them a problem. Triangular polygons will also be more successful under Displacement Maps, which 'raise' or 'lower' surface polygons according to the differences in greyscale colour density. In animation work, all objects undergoing any form of movement are best *Tripled* before rendering commences.

Subdiv

Subdiv (*Subdivide*) divides triangular and quadrangular polygons into smaller triangles/quadrangles. The command will not function on polygons with five or more edges. To ensure this command will always work, use the *Triple* command first. This ensures that *all* polygons are triangular. *Subdiv* is accompanied by a pop up panel which allows you to select a *Faceted*, *Smooth* or *Metaform* subdivision. The command is broadly useful for adding details to objects. The smaller the triangles/quadrangles the finer is the detail that can be incorporated into a surface.

Subdivide / Faceted will insert points along the polygon plane that lies between any two points. Therefore, the newly created polygon faces will maintain the faceted appearance of the original object. Use this on any flat or angular objects you wish to give a more detailed appearance.

Subdivide / Smooth will approximate points along the profile of the object and attempt to maintain any curvature the surface already has. Use it on objects which are curved or rounded where you want to apply a more detailed surface appearance. This option also requires data on the *Maximum Smoothing Angle* (*MSA*) *t*o be used in the process. See below for more information.

Note that this tool can cause tearing of polygons at their join. If this happens, use *Undo*, then perform a *Merge Points/Automatic*. This will ensure further *Subdivide/Smooths* will work as expected.

Maximum Smoothing Angle (*MSA*) defines the maximum angle that can be smoothed across by tools like *Subdivide/Smooth*. It is helpful to visualise the *MSA* as the angle between any two adjacent *Normals* (extended until they intersect) which the Modeler considers before applying any *Smoothing* action. Two polygons whose *Normals* are inclined to one another at an angle *greater* than the *MSA* retain a sharp edge, rather than the join being smoothed over. The *MSA* default setting is 89.5°. This means that any polygons inclined at 90° to each other will remain at right angles after smoothing is applied. An alternative view of the *MSA* is provided by the *Smoothing Threshold*© concept. The *MSA* and the *Smoothing Threshold*© are discussed in detail in Tutorial 2.

Subdivide / Metaform is a special form of subdivide which splits triangular and quadrangular polygons into several smaller polygons. For triangles, the points of subdivision are between each vertex and the geometric centre of the polygon. With quadrangles, the subdivision occurs between the centre of each edge and the geometric centre. At the same time, *Metaform* will smooth around the edges where polygons lie in different planes. *Polygons*, lying along an edge become divided and curved when *Metaformed.* The closer their *Points*, the tighter is the curvature. A cube, for example, will loose its corners and edges to become a much smoother shape. It will look as if it has been sandpapered. *Metaform* can be used in conjunction with *Bevel*, to create new structural features.

As well as smoothing, *Metaform* can also be made to introduce surface jitter. The extent of this is controlled by the *Fractal* setting. The default value of zero introduces no jitter and the *Polygons* are simply divided and smoothed. The default is the most useful setting for creating smooth organic shapes from simpler objects.

The *Metaform* algorithm may be applied more than once to generate very large numbers of *Polygons*. However, memory requirements accelerate rapidly beyond two recursions. There is also the possibility of creating large numbers of overlapping points, leading to object handling errors unless they are *Merged* between each *Metaform* stage.

Also see Tutorial 6.

Align

Align will attempt to make all selected polygons face in the same direction. Thus, it tries to turn all *Normals* to face the same direction. Select the entire object or a part where there are polygons facing the wrong direction and click the *Align* button. This should solve the problem, but it may cause all the *Polygons* to face the wrong direction. However, these polygons are now all facing the *same* direction. In such cases, simply click the *Flip* button. *Align* is complex and not foolproof. Sometimes there are groups of polygons that will not align properly. You may need to *Merge Points* first, because the status of a *Polygon* is dependent on the points that comprise it.

Unify

Unify converts double-sided *Polygons* into single-sided *Polygons*. Select the whole object or a portion containing double-sided polygons and click the *Unify* button. Sometimes, complex objects fool the *Unify* algorithms and will be given a number of single-sided polygons facing the wrong direction. In this case, use the *Align* and *Flip* commands to sort them out.

Flip

Flip reverses the direction of the surface *Normals* from single-sided polygons. It therefore flips the direction from which the polygons can be seen when rendered. Select one or more polygons or objects and click the *Flip* button. *Flip* can also be used to swap the *Starting Point* of *Curves* from one end to the other. *Flip* has no effect on double-sided polygons.

The Tools Menu

The *Tools* menu provides commands which operate on *Objects*, *Points* or *Curves*, so they are grouped accordingly. The *Objects* commands include 2D, 3D and *Boolean* drill operations. Each of the drilling processes require a *Front* and a *Back* layer. The *Front* layer always contains the object to be drilled. The *Back* layer always contains the drilling object. The *Points* group includes commands that merge, align and manipulate points in a variety of ways. The *Curve* functions include the creation of curves and surface paths.

The Objects Group

Drill

Drill (*Template Drill*) takes a 2D polygon (or set of polygons) placed in a *Back* layer (the template) and uses it to slice through the contents of the *Front* layer. The *Front* layer can be a 2D or a 3D object. *Drill* will not work properly if the *Back* layer is a 3D object. If you attempt to use a combination of single-sided and double-sided polygons, and/or a combination of 2D and 3D objects as the template, *Drill* will only operate using the available single-sided, 2D polygons.

When viewed along the drilling axis, the two objects must overlap one another, either partially or fully. After arranging the objects in different layers as described, press *Drill*. A *Numeric* data panel pops up in which you enter the required parameters for the drilling operation. There are four types of *Drill* as described below. The drilling procedure will take place along the ***X***, ***Y*** or ***Z*** axis as determined by your selection under *Axis*. Note that the drilling process does not create an inside surface along the cut edges.

Core

Core provides an object something like an apple core. It is a 'plug' that would fit the hole in the drilled object. Its cross-section is shaped like the drilling object and its ends are shaped according to the surface contours of the *Front* object . It does not, however, have any new

surfaces between the 'ends'. Only those which originate from the *Front* object will be contained in the 'core'.

Tunnel

Tunnel provides the result of drilling a hole into the *Front* object, where the profile of the hole is the profile of the drilling object along the drilling axis. However, the hole will not contain any new polygons within its walls

Slice

Slice is analogous to cutting through an apple with a knife, without separating the halves. You do not remove any part of the object , but you do create new edges where the two pieces were joined. All the polygons remain, but new polygons and edges are created where the blade' template overlapped and passed through the *Front* object. *Slice* leaves all polygons that were within the template area and those that were outside the template area in place. All new polygons retain the *Surface* name of the polygons that were sliced.

Stencil

Stencil is a special form of the *Slice* option. It performs exactly the same operation on the polygons except that the newly created polygons may be given a different *Surface* name from those of the original *Front* object. To enable you to do this, a *Stencil Surface* dialogue box pops up when the *Stencil* command is invoked.

S Drill

S Drill (*Solid Drill*) takes a 3D solid object in the *Back* layer and slices the contents of the *Front* layer with it. This is analogous to *Template Drill*, except that the cutting edges belong to a 3D solid rather than a 2D template projected into 3D. *S Drill* will not work properly if the *Back* layer is a 2D object. The drilling process will not create an inside surface along the cut edges. When viewed along the drilling axis, the two objects must overlap one another, either partially or fully. The drilling object must be a closed 3D solid. For example, a sphere is a closed solid, but a sphere with a hole through it is not and will not work . When the layers have been arranged as described, press *S Drill*. A lister with four *Solid Drill* options will pop up.

Select the *Operation* you wish to perform according to the following:

S Drill/Core

Core provides an object something like an apple core. It is a 'plug' that would fit the hole in the drilled object. Its cross-section is shaped like the drilling object and its ends are shaped according to the surface contours of the *Front* object . It does not, however, have any new surfaces between the 'ends'. Only those which originate from the *Front* object will be continued in the 'core'.

S Drill/Tunnel

Tunnel provides the result of drilling a hole into the *Front* object, where the profile of the hole is the profile of the drilling object along the drilling axis. However, the hole will not contain any new polygons within its walls.

S Drill/Slice

Slice is analogous to cutting through an apple with a knife, without separating the halves. You do not remove any part of the object , but you do create new edges where the two pieces were joined. All the polygons remain, but new polygons and edges are created where the cutting object overlapped. It's like tracing the outlines of the cutting polygons into the polygons of the *Front* object. All new polygons retain the *Surface* name of the polygons that were sliced. The amount of overlap between objects in the *Front* and *Back* layers is important. If the overlap extends completely through the *Front* object, then all polygons are involved. If only the 'front' of the *Front* object is overlapped, then only those 'front' polygons will be affected.

S Drill/Stencil

Stencil is a special form of the *Slice* option. It performs exactly the same operation on the polygons except that the newly created polygons may be given a different *Surface* name from those of the original *Front* object. To enable you to do this, a *Stencil Surface* dialogue box pops up when the *Stencil* command is invoked.

Boolean

Boolean (*CSG Boolean or Constructive Solid Geometry Boolean*) operations will merge objects together, split objects apart, carve objects away, and join objects in a variety of ways. You can model difficult 3D objects with these simple operations. All the manipulations require a 2D or 3D polygon object as the drill in a *Back* layer and the object to be drilled in the *Front* layer. In order for *Boolean* operations to work, the objects in *Front* and *Back* layers must occupy common 3D space. Unlike *Template* or *Solid Drill*, *Boolean* drilling operates in three dimensions, so they should overlap to some degree in all three *Edit Windows*. There are four types of Boolean drill as described below.

Boolean/Union

Union joins the *Front* object and the *Back* object together to form a single combination object. Interior faces within the combination are removed. No new polygons or surfaces are created and all *Surfaces* retain their original names.
Both *Front* and *Back* objects must be closed 3D solids.
The amount of overlap between the *Front* and *Back* objects is significant. If the overlap extends completely through the *Front* object, then all its polygons are involved. However, if the overlap is only over part of the *Front* object, then only polygons in that part are involved. If there is no overlap, the operation will not function.

Boolean/Intersect

Intersect examines the two layers and leaves behind only that portion common to both. It leaves behind only those portions of the *Front* and *Back* layers that overlapped.
(It corresponds to the portion which the *Union* operation discards)
Both *Front* and *Back* objects must be closed 3D solids.
The amount of overlap between the *Front* and *Back* objects is significant. If the overlap extends completely through the *Front* object, then all its polygons are involved. However, if the overlap is only over part of the *Front* object, then only polygons in that part are involved. If there is no overlap, the operation will not function.
No new polygons or surfaces are created, so all *Surfaces* retain their original names.

Boolean/Subtract

Subtract removes the *Back* object from the *Front* object and leaves a copy of itself embedded therein. Essentially, it carves or excavates the *Back* object out of the *Front* object, creating new inside *Surfaces*. These new *Surfaces* adopt the name of the *Surface*s which created them.
Both *Front* and *Back* objects must be closed 3D solids.
The amount of overlap between the *Front* and *Back* objects is significant. If the overlap extends completely through the *Front* object, then all its polygons are involved. However, if the overlap is only over part of the *Front* object, then only polygons in that part are involved. If there is no overlap, the operation will not function.
The results of the *Boolean Subtract* process will vary, depending upon whether the *Back* polygons are single-sided or double-sided.

Boolean/Add

Add joins the *Back* object with the *Front* object so that the overlapping shapes are fully combined into one new object. This is a different process from simply moving the *Back* object into the *Front* and saving the overlapping objects. *Add* causes polygons which are physically overlapping to *Merge* in a sort of mutual drill operation. No new *Surfaces* are created, though some polygons may be subdivided. All *Surfaces* retain their original names. *Add* will work for any combination of 2D and 3D objects.

An insight to the *Boolean Operator* is given in *Tutorial 7*.

The Points Group

Merge

Merge causes overlapping *Points* to merge together into a single *Point*. The resulting *Point* belongs to the same *Polygons* as the *Points* that created it. To operate the *Merge* command, move the relevant *Points/Polygons* so that the *Points* superimpose as closely as possible in all three *Edit Windows*. Ensure that all the items involved are selected and click on the *Merge* button. In the panel which pops up, select the merging procedure and enter any additional values. Click OK. The options for Merge are as follows:

Merge / Automatic

will merge all points that superimpose (more or less) in each window.
Use *Automatic* when dealing with objects or portions of objects that you have cut off a main object, with the intention of merging back later into their original positions.

Merge / Fractional

requires data input which is actually the power of 10. This setting takes the size of the object, then takes a fraction of that size as the *Merge* distance. 'Overlapping' points must be within this distance of one another to be merged.

Merge / Absolute

requests that you enter an absolute distance between points that will be able to merge. The *Merge* command chooses one point as a base point and other points are moved to that location before being merged. Polygons associated with moved points are stretched accordingly. *Merge* does not average out the distance between the points involve. If you need to merge two points, but you are unsure of their distance apart, use the *Absolute* value of 0. This will cause the points to merge. Be careful not to select more than two points, because all selected points are merged by this option.

Merge will not handle points in different layers. Objects or polygons must be in the same layer. *Merge* will only handle the points in selected items. This can be useful if there are many overlapping points, but you want to *Merge* specific ones.

Weld

Weld averages the coordinates of a selected group of points and combines them into a single point, located at the averaged location. Polygons will be stretched to accommodate the welding process. Select two or more *Points* and click on *Weld*. Later editions of the *Modeler* software may take the location of the first or last selected *Point* as the location of the *Weld*.

Quantiz

Quantiz (Quantize) snaps *Points* to the nearest (X, Y, Z) co-ordinate of your choice. Select a number of *Points* and click *Quantiz*. In the numeric panel which pops up, enter the coordinates spacing to use.
For example, X 1, Y 2, Z 3 metres will cause every selected *Point* to snap (move) to the nearest 1-metre X axis, 2-metre Y axis and 3-metre Z axis point intersection. This command is generally employed to move points so that they form planar polygons, or to manipulate them in very small increments for precise positioning. You may find it useful for objects you intend to morph.

Jitter

Jitter moves *Points* randomly within a specified radius, randomising or roughing up the surface of the object. After selecting a group of points, polygon or an entire object, press the *Jitter* button and enter a jitter *Radius* in the panel which pops up. A *Radius* value of zero is 'no jitter', meaning no change will occur. Positive or negative values are acceptable. Select *OK* and the randomisation will occur along each *Axis* specified in the *Radius* field. *Jitter* is used to create a physically rough surface as distinct from applying an optical roughening effect via the *Surfaces* control panel in *Layout*. You might consider using the *Subdivide* command before applying *Jitter* to increase detail.

Smooth

Smooth will move the *Points* in an object to reduce its jaggedness. Select the relevant points, polygons or object and press the *Smooth* button. Enter the *Strength* and number of *Iterations* you wish to apply in the smoothing process. The *Strength* parameter determines the power of the smoothing force. Low values give subtle effects, while high values are more dramatic. The smoothing process employs repetitive calculations or iterations. Enter the number of *Iterations* required. Three is typical. Applying *Smooth* with three iterations is equivalent to applying it three times with one iteration. *Smooth* is best applied to complete objects, but it may be applied to selected parts using the three selection modes.

Set Val

Set Val (*Set Value*) will move selected *Points* along one axis only, to the co-ordinate you specify in the pop up panel. Select the *Axis* along which you wish to move the selected *Points*. Set the *Value* you wish the *Points* to move to along the selected *Axis*. Set Val is an advanced modelling tool that's useful when you want the points of an object to be at an exact location or alignment.

The Curves Group

Make

Make (*Make Curve*) will create a *Spline Curve*. Create two or more points in 3D space and select them sequentially (i.e. *do not* use mass selection tools like *Lasso*). The first point selected will be made the *Starting* point and the *Curve* will follow the points in their order of selection. Press the *Make* button and the *Spline* is

calculated and a *Curve* will be drawn. The more *Points* (*Knots) you* use for the *Curve*, the more segments it will ultimately contain following a *Multiply* operation. *Multiply* is also affected by the *Curve Division* setting that is activated under the *Objects/Options* command.

Make Cl

Make Cl (*Make Closed Curve*) creates a *Closed Curve* in which the *Start* and *End Points* are connected. Select three or more *Points* and press the *Make Cl* button. The greater the number of *Points* (*Knots*) involved in creating the *Closed Curve*, the more segments it will ultimately contain following a *Multiply* operation. *Multiply* is also affected by the *Curve Division* setting that is activated under the *Objects/Options* command.

Start CP

Start CP (*Starting Continuity Point*) enables you to use the *Start Point* of a *Curve* to control the continuity of the curve at the next *Knot*. Select the *Curve* and press the *Start CP* button. The leading segment of the *Curve*, between the first *Point* and the second, will become a broken line. This informs you that the end *Point* is now a *Continuity Control Point*. Go to the *Move/Drag* command and *Drag* the *Control Point* around to adjust the curvature of the polygon *Curve* at the next *Knot*. This operation changes the end points into control points, which are no longer part of the *Spline Curve* proper, but curvature adjustment aids. They will not create *Polygons* when the *Curve* is used for that purpose. Continuity controls are useful when you wish to *Clone* or *Lathe* a P*olygon Curve* in a way that requires a perfectly smooth and uninterrupted join around the central axis. The construction of a wine glass is an example where a continuity point will prevent a dimple being formed in the bottom of the bowl when the curve of the wine glass is lathed.

End CP

End CP (*Ending Continuity Point) enables* you to use the *End Point* of a curve as a control for the curvature of the *Curve* at the next *Knot*. Select the *Curve* and press the *End CP* button. The trailing segment of the *Curve*, between the last *Point* and the next to last, will become a broken line. This informs you that the end *Point* is now a *Continuity Control Point*. Go to the *Move/Drag* command and *Drag* the *Control Point* around to adjust the curvature of the polygon *Curve* at the next *Knot*. This operation changes the end points into control points, which are no longer part of the *Spline Curve* proper, but curvature adjustment aids. They will not create *Polygons* when the *Curve* is used for that purpose. Continuity controls are useful when you wish to *Clone* or *Lathe* a P*olygon Curve* in a way that requires a perfectly smooth and uninterrupted join around the central axis. The construction of a wine glass is an example where a continuity point will prevent a dimple being formed in the bottom of the bowl when the curve of the wine glass is lathed.

Freeze

Freeze instantly converts selected *Curves* into *Polygons*. Select one or more *Curves* (Closed or Open) and press the *Freeze* button, whereupon they will become new *Polygons*.

Smooth

Smooth (*Smooth Curve*) is used to fix any distortion around the 'join' where two *Curves* have been *Merged*. If you move an end *Point* (*Knot*) of a *Curve* into a position superimposing that of a second *Curve*, the *Points* can be *Merged* (*Modify*/*Merge* command) to produce a hybrid *Curve*. However, the 'join' will usually be angular rather than a natural-looking curve. *Smooth* will adjust the curvature of the two components at this point. If the *Smooth* operation is successful, you will see *Starting* and *Ending Continuity Points* appear automatically, along with their broken line indicators.

The Display Menu

This menu contains controls which allow you to adjust the screen display in various ways. You can also get specific information and statistics about the current object.

Magnify

Magnify allows you to zoom in closer to, or to zoom out further from, the current object. It does not resize the object, but rather alters your point of view, so that you are placed either closer to, or further away from, the object. When you click the *Magnify* button, the cursor changes to a magnifying glass. Place the cursor to any relevant position in one of the *Edit Windows* and drag the mouse (LMB down) to the left (zoom out) or right (zoom in). *Zoom* will continue to focus on the portion of the screen directly under the cursor when you begin to drag the mouse. This is different from the *Display/In* and *Display/Out* buttons (see below).

Pan

Pan slides the complete *Edit Window* and its contents left, right , up or down. Click on *Pan* and drag the mouse (LMB) inside a window. The window and its contents will slide in the direction you drag. The *Pan* mechanism can be invoked at any time without using this particular button. Simply hold down the *Alt* key and drag the mouse in a window. The current tool is over-ridden by the *Alt* key.

Measure

Measure allows you to measure the distance between two points in an *Edit Window*. Click the *Measure* button and the cursor changes into two callipers surrounding a dot. Move the dot to the first reference point on the screen and click down either mouse button. A grey cross-hair is placed at the first point . Holding down the mouse button, drag the cursor to any other destination point. As you do this, a yellow line is drawn between the two points and a yellow cross-hair follows the cursor. At the same time, the coordinates window will interactively display the distance between the starting point and the current cursor position. Measure will not work in the *Preview Window*.

In

In is used to allow your viewpoint to move closer into the *Edit Window*. Re-clicking on the button repeats the process in single grid-square steps.

Out

Out is used to allow your viewpoint to move further away from the *Edit Window*. Re-clicking on the button repeats the process in single grid-square steps.

Fit All

Fit All fits all items in all the visible layers to the *Edit Windows*. The *Fit All* command will cause the windows on screen to be zoomed to the maximum size that will fit the objects on screen in all three views. An extended keyboard command is Fit *Window* (Ctrl+a). This will fit the selected item to the *Edit Window* in which the cursor is currently located.

Fit Sel

Fit Sel (*Fit Selected*) will fit the selected item(s) into all *Edit Windows* (*Front Layers* only). The *Fit Sel* command will cause the windows on screen to be zoomed to the maximum size that will fit the objects on screen in all three views. An extended keyboard command is Fit *Window* (Ctrl+a). This will fit the selected item to the *Edit Window* in which the cursor is currently located.

Options

Options provides a pop up panel in which you can make adjustments to the display features of the *Modeler* screen. Note that certain *Options* for the Layout display are set here also.

Orientation governs how the three *Edit Windows* are displayed on screen. You will always have the four window display, but sometimes the location and orientation of the windows can help you to better visualise an object as you design it.

Orientation / ***Logo (XY)*** provides *Top*, *Face* and *Left* views that correspond to *Layout's* XZ, XY and *ZY* views.

Orientation / ***Map (XZ)*** provides *Map*, *Left* and *Back* views. For designing a map, the main view (upper left window) displays the object best.

Orientation / ***Side (ZY)*** provides *Top*, *Left* and *Front* views. The main view is a side view rather than the top view.

Orientation / ***Vehicle*** provides *Top*, *Front* and *Right* views. The *Front* view has you looking into the negative Z axis. The main view is the *Top* view, so if the object were an aeroplane, it would fly to the right of the screen.

The ***Preview*** appears within a circular outline in the *Preview Window*. This circle is a virtual trackball which allows you to position an image of the object independently of its position in the *Edit Windows*. To rotate the object, drag the mouse in the *Preview Window.*

Preview / ***None*** turns off the *Preview Window*.

Preview / ***Static*** displays a non-moving wireframe view of the object.
In *select Point* or *Polygon* mode you can select points or polygons by clicking on them in the *Preview Window*.
In select *Volume* mode, you can select points or polygons using a volume box in the *Preview Window.*

Preview / ***Moving*** displays an oscillating image of the object.
Selecting the *Moving* option pops up two further options which determine the type of moving image displayed.

Moving / ***Wire*** displays a moving wireframe.

Moving / ***Solid*** displays a moving solid rendition.
In either case, a short time is required for the 'frames' of the moving animation to be computed. This is also the case whenever the virtual trackball is operated.

Visibility turns on or off the visibility of various aspects of the *Modeler* interface. This option set is fairly self-explanatory. A tick mark in any box indicates that the option is active.

Visibility / ***Points*** toggles visibility of *Points*.

Visibility / ***Faces*** toggles visibility of *Faces* (*Polygon faces*)

Visibility / ***Curves*** toggles visibility of *Curves*

Visibility / ***Normals*** toggles visibility of *Surface Normals*

Visibility / ***Grid*** toggles visibility of the *grid* and *axis labels*

Visibility / ***Backdrop*** toggles visibility of any available *Backdrop Image*

Unit System determines what units of measurement are used and displayed by the *Modeler* screen. The selected units are also employed in Layout for the *Grid* display.

SI units are the *Standard International* units (Gigametres, megametres, kilometres, metres, millimetres, micrometres and nanometres)

Metric units are those of the *Metric System*. This is identical to the *SI* system, with the addition of centimetres.

English units are *English Imperial Measures* and used miles, yards, feet and inches.

It is recommended that the metric system be used since it is assumed throughout this guide.

Grid Units affect several parts of the interface. The grid display, the on-screen graticule and the zoom step quantity are affected by these parameters. The parameters have no effect on the size of any object, only the grid placement display. You will find that 1 2.5 5 and 1 2 5 are your most commonly used settings .

Grid 1 provides a square grid 1 unit in size. If the grid size is 1 metre, grid lines fall every metre. The grid resizes in values that begin with 1, as in
10m--1m--100mm--10mm--1mm--etc.

Grid 1 5 provides a grid 1/2 unit in size. If the grid size is set to 1 metre, grid lines fall every half metre and at every metre. The grid resizes in steps that begin with 1 and 5, as in
1m--500mm--100mm-- 50mm--10mm--etc.

Grid 1 2.5 5 will resize the grid in steps beginning with 1, 2.5 and 5, as in
1m--500mm--250mm--100mm--50mm--25mm--10mm--etc.

Grid 1 2 5 shows steps that begin 1, 2 and 5, as in
1m--500mm--200mm--100mm--20mm--10mm--etc.

Grid 1 2 shows steps that begin with 1 and 2, as in
1m--200mm--100mm--20mm--10mm--etc.

Grid Snap forces *Point* creation and object movement to be limited to specific increments.

Setting a fixed *Grid Snap* also forces the grid's on screen graticule to be displayed at that increment.

None unlocks the snap, so the cursor moves freely and is unconstrained by the grid intersections.

Standard grid snap is set at 1/10 of the current grid size.

Fixed grid snap allows you to enter a snap value in the pop up box. The grid will resize itself to the intersect at the value you specify here.

Note: Any settings changed in the *Display Options* panel are *not saved* when you exit or quit from Modeler.

BG Image

BG Image (*Background Image*) allows you to display an *Image* (previously loaded into Layout) as a *Background Image* on the grid of an *Edit Window*. This facility is useful for ensuring that colour maps, secularity maps, etc. 'fit' the surface of the intended object in a satisfactory way. You can also use *BG Image* to check how an image map might be sized by the *Automatic Sizing* function in Layout's texture mapping controls.

Note: *Backdrop Visibility* must be on (see *Display/OptionsVisibility/Backdrop*)
One or more *Images* must be pre-loaded into Layout because Modeler takes its image list from the corresponding list in Layout. Click on the *BG Image* button and a panel pops up on which the required *Background Image* can be selected. If there are several images available, the current *Image* name is displayed in the scroll bar. Clicking on the bar with the LMB will display the full list, including the option to display *None*. The list may be extensive and continue beyond the visible panel. If so, an *up* or *down* arrow will indicate that further names are available. Loaded *Images* are stored in chip RAM so they must be relatively small or few in number. Move the cursor to the required *Image* and release the LMB. Click *OK* and the image (in low-res grey-scale) will be displayed in an *Edit Window*)

Select the required projection *Axis* for the image. This determines which *Edit Window* will contain the image. If you are using Modeler's default *Logo Display* setting, then:

Axis X will place the image in the *Left* view
Axis Y will place the image in the *Top* view
Axis Z will place the image in the *Face* view

The ***Centre*** parameter allows you to place the image at any specific coordinates in a window. Enter the X, Y, Z values of the location and the image will be centred on that point

The ***Size*** parameter allows you to resize or stretch an image if it was produced in a screen aspect ratio different from LightWave's 1:1 (square pixel) format. For example, images captured from video are produced at an aspect ratio of 4:3 as used by television displays. In this case, the image would need to be stretched horizontally by a factor of 1.3 to display the image in its 'normal' state. If the image has been scanned or transferred from a system using square pixels, the sizing parameters should all be left at 1.

Click the ***Automatic Size*** button to automatically size the image to fit the largest selected object in the current layer. This is analogous to the *Automatic Sizing* button in Layout's texture mapping controls and can be used to check how the image will 'fit' the object when planar mapping is used.

The ***Units*** parameter set the unit of measurement for the operation.

The ***Dark-Light*** slider will let you change the contrast of the *BG Image* which may improve its visibility.

Invert will invert the brightness values of the *BG Image* (make it negative) which may improve the visibility of detail.

Note: If a *BG Image* does not appear after setting it up in the pop up panel, check that *Backdrop Visibility* is ticked on. (See the *Display/Options/Visibility/Backdrop* button).
Alternatively, check the *Grid* size. Images default to a specific size and if you are zoomed out too far, you may not be able to see the image.

The Selection Group

Stats

Stats (*Statistics*) is used to get information about points, polygons or the contents of a volume, displayed in the current layer. With the *Stats* command you can select/deselect items based on point count, polygon count,

surface names, volume inclusion and volume exclusion. You can also use it to locate non-planar polygons. There are three *Stats* menus, dealing with *Points*, *Polygons* and *Volumes*. Click the appropriate mode selection button (*Point*, *Polygon* or *Volume*) and then click on the *Stats* button. To get statistics on each element of the view, you must select the required mode first. To change mode, you must exit the *Stats* menu, change the mode and go back into the *Stats* menu.

Each select mode pops up a different *Statistics* panel when you click on the *Stats* button. All three types of statistics menus are categorised. Click on the (+) button beside the item type you wish to select. Click on the (-) button beside the items you want to deselect. The selection will be accepted and you will be returned to the edit screen. If you want to make further selections, click the *Stats* button again.

Points Statistics panel gives you information about all the active *Points* in the current layer. The *Total Points* count is categorised according to the number of *Polygons* they are shared by For example, *Points* shared by *Zero Polygons* are free points in space and should probably be cut from the object. Any *Points* shared by only *One Polygon* will be located on the edge of a mesh. *Points* shared by *Three*, *Four* or *More than Four Polygons* will be located inside a mesh.

Click the (+) button against any category to select those *Points* in the *Edit Windows*. The *Statistics* panel will close when you click *OK* and the requested *Points* will appear highlighted. To deselect the highlighted *Points*, either :

Go back to the *Statistics* panel (via the *Stats* button), and click the (-) button alongside the relevant *Points*, or
Simply click on the *Points* mode selection button with the RMB.

Polygon Statistics panel gives information about the types of active *Polygons* present in the current layer. It allows you to select and deselect them according to a number of different attributes. You can deal with *Polygons*; *Faces*; *Curves*; *Polygons* containing 1-4 or more *Vertices*; *Polygons* belonging to a specific *Surface* name; or Non-planar *Polygons* (based on the *Flatness Limit* - found in the *Objects/Options* menu).

Click on the (+) button beside the type of *Polygons* you wish to select, or the (-) button to deselect them.

To select *Polygons* by ***Surface*** name, place the cursor on the scroll bar near the bottom of the panel. Click it with the RMB and whilst holding down the RMB, scroll down the list to locate the *Surface* name required. The list may be extensive and continue beyond the visible panel. If so, an *up* or *down* arrow will indicate that further names are available. Release the RMB and click on the (+) button beside the entry '*with Surface'*. The required *Polygons* will be highlighted in the *Edit Windows.*
If you need to select more Polygons, you can use Statistics repeatedly to add to the selected group.

To locate/select/deselect ***Non-planar Polygons***, click the (+) button on the very bottom row of the *Statistics* panel. The panel will close and any *non-planar Polygons* will be highlighted.
By default, Modeler considers any *Polygon* more than 2% out of alignment to be non-planar. You can alter this default setting using the *Flatness Limit* setting in the *Objects/Options* menu.
Click the (+) or (-) button to select, or deselect the *Non-planar Polygons*.

The ***Volume Statistics*** panel allows you to select or deselect a number of *Points* or a number of *Polygons* which are located *Inside a Volume* box or *Outside a Volume* box.

First, click the *Volume* selection button (ensure you are in *Include* or *Exclude* mode, whichever is appropriate) and then define a boundary *Volume*. Click on the *Stats* button to bring up the *Volume Statistics* panel. Click on the (+) or (-) button to select or deselect the relevant items.
To display the result of your selection, click on the *Points* mode button to display selected *Points* or click on the *Polygon* mode button to display the selected *Polygons*.
The quickest way to deselect the *Points* or *Polygons* highlighted by this process is to click on the *Points* or *Polygon* mode button with the RMB.

Info

Info (*Information*) provides information about *Points* and *Polygons*. It also allows you to place *Points* into specific locations, to check *Polygon* integrity, and provides a method of converting *Double-sided Polygons* into

Single-sided Polygons. This command requires you to formally select the relevant items. This is one of the commands where the (*none* selected = *all* selected) principle does *not* operate.
To work on *Points*, first click on the *Points* mode button. Select one or more *Points* and then click the *Info* button. The pop up panel displays information about the *Points* in their order of selection. Click on the *Next* or *Previous* button to move to the next or previously selected *Point*.

You can change the location of the current point by inserting fresh coordinates in the X, Y, Z fields and clicking the *OK* button (or hitting the *Return* key). The *Point* is instantly relocated. You can change the coordinates of several *Points* before pressing the *OK* button or *Return* key. Just click on *Next* or *Previous* and insert the required coordinates of all the *Points* you wish to change before clicking *OK* or hitting *Return*.

To work on *Polygons*, first click on the *Polygon* mode button, select the relevant *Polygons* and then click on the *Info* button. The pop up panel displays information about the *Polygons* including their *Type*, *Number of Points* (*vertices*), *Surface* name and their *% Flatness.* The data are presented in numeric order starting with the first *Polygon* selected. Click on the Next or Previous button to move through the list. The current *Polygon* can be deselected by clicking the *Deselect* button.

Sel Conn

Sel Conn (*Select Connected*) automatically selects all *Points* or *Polygons* that are connected to the currently highlighted *Points* or *Polygons*. First click the *Point* or *Polygon* mode button, according to the items you wish to work with. Click *Sel Conn* and all connected items also become highlighted. This command is useful in the same way that selecting by *Surface* name is useful. Any object that is surrounded by others may not be easily selected using a *Volume* and may have several *Surface* names, making selection by that route unwieldy. Select a *Point* or a *Polygon* and click the *Sel Conn* button. This command will select everything that is formally connected. It will not select overlapping *Points* that have not been *Merged*, for example.

Invert

Invert is a useful way of inverting the seleted/unselected status of *Points* or *Polygons*. In either *Point* or *Polygon* mode, clicking *Invert* will reverse the selection status if all items in the current layer. *Selected Points* or *Polygons* will be *deselected* and *unselected Points* or *Polygons* will be selected.

The Visibility Group

Hide Sel

Hide Sel (*Hide Selected*) is used to hide selected *Points* or *Polygons* from the on-screen image. The hidden items are not removed from the object itself, only the *Edit Window* view. In either *Points*, *Polygon* or *Volume* mode, select the relevant items and click on the *Hide Sel* button. The selected items will become hidden. Further items can be added to the hidden group by repeating the above process.

Hide Uns

Hide Uns (*Hide Unselected*) performs the opposite effect to the *Hide Sel* button. It is used to hide all unselected *Points* or *Polygons* from the on-screen image. The hidden items are not removed from the object itself, only the *Edit Window* view. In either *Points*, *Polygon* or *Volume* mode, select the relevant items and click on the *Hide Uns* button. All unselected items will become hidden. Further items can be added to the hidden group by deselecting the remaining *Points* or *Polygons* and then re-selecting a further group. Press the *Hide Uns* button to add the unselected items to the previous group.

Unhide

Unhide will return all hidden items to view after they have been hidden using *Hide Sel* or *Hide Uns* . *Unhide* performs a similar function to *Undo* when only one group of items has been hidden. However, *Undo* will only reverse the last operation, whereas using *Unhide* you can unhide all the items which have been hidden after several hiding processes.

Modeler - A Crash Course

If you don't want to study the detail contained in the earlier sections of ***WaveGuide©***, try this crash course for a quick get-you-going summary on making simple *Objects*. If you don't even want to venture that far, there are plenty of excellent *Objects* already available in the *3D* drawer.

You can generate 3D models from *Primitive* solids like the *Ball, Box* and *Cone*, or start from the very basic points. The *Primitives* are often more trouble than they're worth and you can quickly get the results you want from scratch. However, unless you wish to develop something very special, it is probably easier to import pre-constructed models. These can be modified to suit your needs within *Modeler* of course. Thankfully, there are dozens of models already available in the *3D/Objects* drawer, along with some pre-organised *Scenes* to render and animate. You can also get them off the Internet and from the public domain.

As an example, to create a sphere or spheroid to render as a planet, etc., first press the *Objects* button. Go down to *Ball* and press this. The pointer changes to a ball. On one of the *Edit Windows*, drag out a bounding box with the mouse. You will get a circle of sorts. It will not have any third dimension until you drag the bounding box out on the third view, which is presently just a line. When it all looks about right, press *Make*. You'll get a ball which may not look very impressive at this stage. It may not be spherical and it will probably be too coarse due to the small number of facets (polygons). This is easily fixed. Click the *New* button and let's start again.

Click *Ball* and then click *Numerical* to get a pop up panel in which you can select the *Ball Type* and enter sizing and positioning data. *Ball Type* provides two alternatives, which differ in the way the surfaces are split into *Polygons*. The *Tessellation* type produces a much smoother surface when rendered. If that isn't too important, select *Globe* type. You'll see a significant difference between the two types when each type is generated on screen. The geometric centre of the sphere is positioned by entering its (X, Y, Z) coordinates in the *Centre* fields. You can control the relative size of the sphere by increasing or decreasing the *Units* of dimension to be

used. These are *nm* (nanometres), *um* (micrometres/microns), *mm* (millimetres), *cm* (centimetres), *m* (metres), *km* (kilometres), *Mm* (Megametres) and *Gm* (Gigametres). All *Objects* are saved with their scaling data. Thus, when several *Objects* are loaded into the same *Scene*, their size relative to each other and most importantly, to the *Grid* size, are accurately drawn.

To get a perfect sphere, enter identical dimensions in the three *Radii* fields (X, Y, Z). Alternatively, you can constrain the cursor to draw perfect circles by holding down the Shift key as you drag out the bounding box. In the panel, you should enter the required number of vertical 'slices' and horizontal 'layers', which divide up the sphere into facets. So, in the *Sides* and *Segments* fields use numbers like 20, 30 or 50 to increase the number of *Polygons*. When all the numeric data are as you want them, click *OK* and the panel will close to leave the bounding box for the latent sphere. If this looks good, click the *Make* button. The sphere will then be drawn, though the bounding box will remain because you are still in the *Ball* creation mode. Click somewhere in a *Blank Area* to switch it off. Remember, more polygons mean better pictures, but they need a lot more memory. If you wish to save the sphere *Object*, press *Save* and give it an appropriate name and file location.

You cannot add surface texture/colour to *Objects* in the *Modeler* environment. You should save them to disk (*3D/Objects* directory) then load them into *Layout* to add surfacing information via the *Surfaces* button. However, each *Surface* should first be defined and named in *Modeler*. A *Surface* is a pre-determined collection of polygons and requires its own name so that a finish can be allocated to the name. There is a dialogue box for assigning names (*Polygon/Surfaces*), so you just enter a logical name for the group of polygons selected. All surfaces given the same name will get the same finish. You can allocate a different finish to any number of different surfaces. If you don't pre-name a surface, it will be labelled *Default* in the requesters you will encounter later. *Surface* polygons may face 'inwards' or 'outwards'. Only outward facing surfaces will render as such. Inward facing surfaces render as 'holes'. You can see which direction a *Surface* faces by checking them in *Modeler*. Details on all this follow.

In *Polygon* mode, click within the mesh of polygons which make up a surface. Those connected directly to the polygon under the cursor will turn yellow (i.e. become highlighted). Pressing the '] ' key will cause all polygons within that same surface to become highlighted. Each surface *Polygon* has a perpendicular line (the *Normal*) pointing in the relevant direction. *Surfaces* usually face outwards by default, though you can easily reverse this by accident or design within *Modeler*. The orientation of all the polygons in the selected surface can be equalised by pressing *Align* button (Polygon menu). Polygons facing the wrong direction can be flipped by pressing *Flip* (*Polygon* menu). You must de-select polygons (or points for that matter) before saving, otherwise only the highlighted bits are saved. Deselect polygons (or points) by clicking on the *Polygon* (or *Points*) button at the bottom of the screen with the right mouse button. You can also use any *Blank Area* for this. Either way, you may first have to deactivate the tool in use by left mouse-clicking its button.

Before saving, you are sometimes wise to give the *Surfac*e of the *Object* a logical name. You can split the *Surface* up into several parts and name each one separately. This allows the *Layout* interface to apply a different finish to each part. To name the whole sphere, select *Polygons* at the top and *Surface* in the *Transform* group. A box pops up to allow you to rename the surface from '*Default*'. To identify a particular part of the surface, you must first select the required polygons. In *Polygon* mode you can select the necessary polygons by holding down the Shift key while you click over the surface with the curser/mouse. Alternatively, draw a lasso around the sphere in one of the 2D views, using the right mouse button. All the enclosed polygons will turn yellow (selected). When the required polygons are all selected, press *Surface* (in the *Transform* group), to bring up the *Change Surface* dialogue box. Enter a suitable name, then save the *Object*. Somewhere in the *3D/Objects* drawer is most the logical place. (This drawer contains a lot of interesting stuff).

You can deselect part of a group of polygons by re-clicking on one of their points with the left mouse button. This allows you to isolate different bits and assign them a different surface effect. To help with this, you can reverse selected and unselected points or polygons using the tools on the left hand side. You can also *Hide* selected (or unselected Polygons) to help out when the object is very complex. When all the different surfaces have been named, re-save the *Object*. When the *Object* is imported into *Layout*, you can apply surface effects to each part using the *Surfaces* interface. After you have applied different surfaces to the different polygon groups, you can save the *Object* again using the *Scene* interface. *Objects* saved via the Layout interface will retain the *Surface* data they currently display. This ensures that the *Surfaces* always accompany the *Object* wherever you use it.

Text Objects for Video Titling, Logos, etc.

Text Objects drawn using computographic fonts (a dozen or so are supplied in the *ToasterFonts* drawer, sub-directory *SoftMaker*) and are good for generating animated logos, etc. These are PostScript Type 1 fonts. *Text Objects* are produced like other objects, using *Modeler,* then imported into *Layout*.

Within *Modeler*, click on *Objects*, then *Text.* This gives the *Generate Text* panel in which the required *Font* is selected and the desired word or phrase is typed into the *Text* field. The current *Font* is shown in the *Font* bar. This is one of a set, which can be accessed by clicking an the bar bearing the name of the current *Font*.

Change the *Font* to another in the list. If the list does not contain the desired *Font*, it can be loaded by clicking *Load*. This will pop up a file requester with the font list for the default *Fonts* directory which is normally the *ToasterFonts/SoftMaker* drawer. See *Mod-config* for details on changing the default directory. Any source of Postscript type 1 fonts may be used. Avoid loading large numbers of fonts when memory is at a premium. Unload any non-required fonts by clicking on *Remove*. The current *Font* will be unloaded from RAM and the next in the list will appear.

See *Objects* menu for details on the *Sharp* and *Buffered* buttons

OK all this and you get a flat, 2D object in the form of the word. To give the object depth, the third dimension is created by *Extrude*-ing the word along the appropriate axis.

First, click *Multiply* and then click *Extrude*. The cursor changes to a red arrow to signify extrude mode. Click this on the *face to be extruded* (i.e. the word object's face, seen in the front view). The face is now contained within a bounding box and the extrusion distance cursor (a yellow T-shaped draw-bar) appears in the top and side views. The thickness of the extrusion is controlled by the moving the T-bar with the mouse (or using the *Numeric* field). When the desired thickness is set, press *Make* and the extrusion is computed.

The edges of your chunky logo can now be given an attractive finish by using the *Bevel* button.

Details on the *Bevel* command can be found in the *Multiply* menu. Use fairly small settings in the requester which appears, otherwise the depth of the bevel may exceed the width of the object and some very strange effects are created. The faces and bevels can be assigned similar or different surface names, so that they can be rendered very professionally, using reflectivity and specularity mapping. There's at least one logo *Scene* example, a 'World Premier' type animation with reflective lettering, already in the *3D* drawer.

Layout - A Crash Course

If you don't want to wade through all the detail contained in the earlier sections of ***WaveGuide©***, try this crash course for a quick get-you-going summary of rendering a *Scene* using the *Objects*, etc. already available in the *3D* drawer.

With the *Edit* group set at *View* and the *Mouse* buttons set at *Move* means that dragging with the LMB will move the *View* as seen from the *Perspective* viewpoint. Try it now. Moving the mouse left and right, front to back. To move up and down, use the RMB. The *Perspective* view responds in real time. Now click on the *Rotate* button under the *Mouse* menu. The LMB and RMB now give rotational control of the *Perspective View*.

Rotational changes are made according to *Heading* (*H*), *Pitch* (*P*) and *Bank* (*B*). These notations are used in flying aircraft and are used by LightWave because both 'you' and the *Camera*, are suspended in space, viewing the scene. In both *Move* and *Rotate* modes, the position of the *View* is indicated in the small black screen at the lower right of the interface. Its title will actually change according to the function selected in the various button banks.

Heading is analogous to compass heading in degrees of rotation from the default zero heading (0^0), looking into the +Z axis. It is controlled by dragging left or right with the LMB. *Pitch* is rotation of an aircraft about the wing axis. It indicates head-up or head-down. *Pitch* is controlled by dragging forwards and backwards with the LMB. *Bank* is rotation around the aircraft's fuselage, in which the left or right wing rises from the horizontal. *Bank* is controlled using the RMB in a left/right direction.

Click on *Camera* in the *Edit* bank. The *Camera* is now highlighted and a dotted line shape appears to emanate from it. The dotted line represents the rectangular field of view seen by the *Camera* in its current internal settings. The *Camera* is now the edit item, awaiting your input. Use the LMB and RMB to *Move* or *Rotate* the

Camera in exactly the same way as you edited the *Perspective* view. When you move the *Camera* to a different location, you will notice that the initial position is marked with a small white cross. This cross is an index mark indicating a *Key Point*. A *Key Point* is the location of the item in the previous *Key Frame*. A *Key Frame* is generated by pressing the *Create Key* button at the bottom of the screen. Every time this button is pressed, the location and rotation values (and many other parameters) of all selected items in the *Scene* are set in memory.

You alter the position of everything in the 3D space using the mouse. The *Edit* buttons allocate mouse control to the selected item. The two *Mouse* buttons alter the function of the actual mouse buttons, giving either linear or rotational motion to the selected item. The left and right mouse buttons translate the selected function into horizontal (left-right or front-back) or vertical (up-down) orientation. Also on the left are buttons to provide additional functions related to the position, size and motion of selected items.

On entering *Layout*, (no *Scene* loaded), the default view is *Perspective*, showing an *XZ* grid plane and the *Camera*. The *Camera* is the 'eye' through which LightWave sees the *Scene*, works out the lighting effects, etc. and records the results. You, however, can view the *Scene* through the *Camera* lens or from other positions determined by the *View* buttons. The *XZ* plane is one slice of the 3D universe within which LightWave *Scenes* are generated. The available universe extends well beyond the visible grid in all directions. The third dimension is up and down relative to the *XZ* plane and is the Y axis (i.e. +Y and -Y), which are above and below the visible grid respectively. To define the position of anything in 3D space, you have control in the three planes *XY*, *XZ* and *ZY*. You can also view the *Scene* from each of these positions by selecting the required one in the *View* set.

To get an initial feel of how all this works, select *Move* in the *Mouse* set, press the left mouse button and drag the mouse left and right or up and down. The view responds in real time. The *X, Y and Z* buttons in the *Mouse* set control in which direction the mouse has effect. Press *Rotate* and the control becomes rotational. The control here is now defined by the three buttons marked *H* (Heading), *P* (Pitch) and *B* (Bank), much like flying an aeroplane. Deselecting (re-click on it) one or more of these three buttons eliminates that orientation from the mouse. The *Zoom Factor* allows the view to be quickly adjusted by zooming in or out according to the directional controls selected. Use the mouse with plenty of movement. This can be useful when you are looking for an object or light which you have placed in the scene, but have 'lost', through positioning problems. You can zoom out until everything placed in the 3D space, including the XZ grid, simply disappears, so any *Objects* which have got pushed well out of sight should easily be found.

The *Edit* set allows you to control various aspects of an *Object*, a *Bone*, a *Light* or the *Camera*. Additional functions may be added to the *Mouse* bank, according to which you select. Most are self-explanatory.

When an *Object is* loaded into Layout, you will see a small cross within the *Bounding Box* that LightWave uses to manipulate it in 3D space. This is the *Object's Pivot Point*. This is the point around which the *Object* will rotate when acted upon by the *Rotate* commands. The *Move Pivot Pt* button allows you to move the point of rotation of a selected *Object* when the *Mouse Rotate* function is selected. It is moved with the mouse. Any *Object* which has been resized or moved or rotated from the as-loaded parameter within Layout must be *Reset* before their *Pivot Point* can be altered.

Selecting *Size* allows you to enlarge or reduce an *Object* to the desired size relative to others. This is done by dragging the mouse, though large changes are better done via the *Numeric Input* button. *Numeric Input* provides a box into which *Object* sizing data can be entered. The *Reset* button reverses the last change in size, position etc. The *Centre* button centres the selected item in the middle of the view screen. This is useful if an item you want to see is slightly out of view.

The button set along the bottom of the screen relates to rendering of scenes and scene animation. The Menu buttons along the top of the screen provide pop up dialogue boxes via which you manipulate dozens of different aspects of the items to which they are assigned. More on these later.

Layout is the interface through which you create a *Scene.* A *Scene* is constructed along the principles of a theatre stage, which can be as big or as small as you wish. The *Scene* may contain just a single picture (a frame) or a series of frames, which provide for animation. On the stage you place optional *Objects, Lights, Backgrounds* etc. and there is always a *Camera,* through which the scene is recorded and by default there is always an *Ambient Light* source. You must also have at least one other *Light*. Each of these buttons provides a dialogue box via which you set down exactly what you need.

All *Objects*, *Lights* and the *Camera* can be moved around the stage using the mouse, after you have selected the respective item in the *Edit* panel and the required move mode in the *Mouse* panel. Selected items are highlighted yellow on the screen and its name appears in the bar below the window. If there are several *Objects* or *Lights* in the *Scene*, you have to pick which one you need using the scroll bar. Left and right mouse buttons give different directional control. Movement of the selected *Object* is controlled by the *Bounding Box*, which temporarily replaces the *Object.* The Bounding Box has an index pointer on one face to allow you to monitor the object's orientation throughout its path.

You can view the set-up from any position occupied by an *Object*, a *Light*, or the *Camera*. You can also get more global views from above, front or sideways. Logically, you can only use the *Camera*'s view to *Render* a *Scene*, but the view from other positions is often useful. These can help to detect unwanted collisions between moving objects for example. The *Perspective* and the other 'external' viewpoints are useful for checking the movement of items which are not necessarily in camera shot.

Select the viewing position using the *View* control panel, then select the *Object, Light, Camera, Perspective, XY*, *XZ* or *ZY* viewpoint. Again, the *Object* and *Light* viewpoints need to be further defined if more than one such item is in the scene. Just scroll through the items till you reach the one you want. The global views can be adjusted to allow you to take very close or very distant views from whatever position you select. You can therefore *Zoom* in or out or *Move* the scene around to centre on some particular item.

You may also alter the relative sizes of *Objects* using the *Size* button. This can be either mouse controlled or via the *Numeric Input* button. Pressing this brings up the sizing box, which allows you to alter the relative scale of the *Object* in one, two or all three axes by punching in a suitable figure. This can be very useful, because pre-prepared *Objects* are not necessarily stored on the same scale. It means that sometimes, an object imported from another directory or floppy looks like a tiny square rather than what you expected. In such a case, entering a large number in all three fields of the scaling box should sort things out. The mouse has a relatively small effect on numerical values, which are constantly displayed at bottom right.

Surfaces

Groups of polygons make up the different *Surfaces* of an *Object*. Unless you define a particular group as a particular surface, LightWave assumes the whole surface is the same and calls it '*Default*'. In *Modeler*, you define a particular group of polygons by selecting them with the mouse (left hand button) using the *Polygon* menu and *Polygon* mode. If you press the shift key while selecting, you can keep adding polygons to previously selected groups. This is useful when you need to zoom into the object to select very fine details. Once a set of polygons are fully selected (yellow), assign this surface a separate name by using the *Surfaces* option in the tool bar. Enter the name in the dialogue box.

One of LightWave's methods for applying surface finish is to use 'transfers', in the form of bitmap 'cut-outs', which are actually brushes (preferably 24-bit quality), which you have to create separately in whichever paint package you favour. You will find it useful to have a screen grabber running in the background when working on surfaces. You can then save an outline or 'cut-out' of your surface to import into the paint package later on.

Load texture brushes or full screen images into LightWave via the *Images* button. You can store as many as memory allows, and you call them up later via the *Surfaces* button.

The surface finish may consist of several elements, like a colour map, a reflectivity/specularity map, height (bump) map etc. This all seems a bit of a chore, but they work very well. They are assigned via the 'T' button (Texture) associated with the surface characteristic being worked on. There are only a few procedural (mathematically computed) textures in version 3.5 (such as *Fractal Noise, Crust* etc.) whereas you get a hundred or more with Imagine. This is the one area where Imagine is superior, because they can be superbly 'organic' and take virtually no memory. The procedural textures are also accessed via the *T* button in the *Surfaces* requester. There is a procedural textures demo scene in the *3D* drawer. Textures can be animated by changing one or more of their settings over time. For example, you can create lifelike flames, explosions, etc. See also the *Images* section.

Lights

Lights come in several forms. The *Ambient* or background lighting comes from an invisible light of which you may control brightness and colour. This light fills in the areas in shadow from other light sources, so a very low intensity will produce densely black shadows. However, other lights can be added and given very powerful properties. The *Lights* button brings up the lights interface. Add a light using the *Add Light* button. *Lights* may be *Distant, Point* or *Spot.* For *Distant* lights you only need worry about their direction, colour and intensity/brightness. *Point* light sources emit light spherically, so you control their position, brightness and colour. *Spot* lights emit a cone of light according to the *Cone Angle*. This type of light needs to point in the correct direction as well. Control this with the mouse after selecting *Lights* and highlighting the relevant one. Select *Rotate* and *Move* according to what you want it to do. The screen will highlight a selected light and make it very clear what sort it is and what direction it points (if it's a spotlight). In all cases, the light intensity can be made to fall off with distance, or not, according to your wishes. You can rename each light to anything you wish for easy reference.

Camera

Camera brings up the Camera control system. It allows various focal lengths and picture formats as well as the rendering quality. One button worth checking out is *Segment Memory*, which allocates a portion of RAM to the render screen. Renders are done in horizontal sections the width of which is determined by the *Segment Memory.* Allowing about 6MB for the background programme, increase *Segment Memory* to as many bytes as you feel you can spare. This permits renders to go at optimum speed. If there's not enough memory to render a segment, the programme will flash up a message about allocation of buffers, etc., but it sometimes just crashes, so be warned.

Antialiasing can be either off or on, with three degrees of refinement. Expect a *considerable* increase in render time with any antialiasing turned on. Visual quality is of course much improved with it on. (Try the 'pop can' animation to see what I mean).

Images

This *Images* interface allows you to load up several IFF frames (any size/depth) or brushes (any size/depth) into LightWave and then use them in a variety of ways. The first is to 'coat' an Object's *Surface* using an *Image* selected from the list. This can be as a colour map, a transparency map, or a specularity map. The latter two operate via the grey-scale parameters of the Image. The *Image* can be wrapped around the *Object* according to its shape. It may be planar (flat), cylindrical, cubic or spherical, according to what you wish to achieve. *Images* can also be used as backgrounds, foregrounds or reflection maps (see *Effects*).

Effects

The *Effects* button brings up a dialogue box with which you can integrate the pre-loaded *Images* into the render. You can also do things with the plain background such as create colour gradients or use an *Image* in the *Background* or *Foreground*. These *Images* may also be used as reflection maps for *Objects* which you have given a highly reflective *Surface*.

Scenes

A *Scene* is a totally composited arrangement of *Objects/Lights/Surfaces/Paths* etc. Whenever you alter anything which affects some aspect of the scene, remember to save it via the *Scene* button and using the resulting requesters. The *Scene Overview* contains lots of useful information about the scene as a whole. Each *Object, Light, Null Object* and *Camera* etc. are listed, together with data about each one. The second column on the left tells you the item's type in the form of a little icon. Clicking the icon will either change its appearance or remove it altogether. This results in a corresponding change in the appearance of the scene on the view screen. No icon in the list means the item will not be shown on the screen. Its position will be marked by its axis only. This is useful if you have a cluttered scene and can't see the wood for the trees as it were. LightWave remembers the item is still there, of course. If you click on an *Object* icon, it's 'window-frame' shape changes to only part of the window-frame. These parts correspond to outline/bounding box, surface points only and surface polygons only. The results are easier to understand by seeing than by describing. They affect the way the object displays during scene construction.

Object Skeleton and Bones

Bones is a system for distorting an object by giving it a 'skeleton' and controlling that instead of the 'surface' of the object. The computer adjusts the 'skin' of the object according to the way you move the skeleton.

The general procedure for using bones is as follows:

Select the relevant *Object* as the *Edit* item
Select the *Objects* menu
Select *Add Bones,* once for each bone you wish to add to the *Object*
    Each *Bone* added in this way is independent of the others.
    Add Child Bone places additional bones onto the first one, much like creating a spinal column. They follow their leader.
Close the *Objects* menu
In Layout, select *Bone* as the *Edit* item
Orientate each *Bone* in relation to the *Object* using *Move* and *Rotate* controls.
Set the physical size of the *Bone* in relation to the *Object* or a part thereof.
    Size a Bone using the *Rest Length* control.
    DO NOT SIZE BONES WITH THE *'SIZE'* CONTROL.
    Use the *Move* and *Rotate* buttons to place the bone where you want it.
    The mouse control when orientating and moving a *Bone* may be different from what you expect. Practice!
When everything is where you want it press 'r'. This tells LightWave that the *Bone* is now *Active*

and in its *Rest Position* (the position in which the *Object* appears in its normal state).
Create a *Key Frame* for each bone individually. If you forget to do this, the *Bones* will jump back to their as-loaded positions, distorting the *Object* as they do so.

Animation

To give you an initial feel, from the *Layout* interface, click on the *Scene* button to open its requester. Click on *Load scene* and select *Space fighters* from the list. Once the pre-arranged scene parameters are loaded, the *Scene Overview* lists the components of the scene and to its right, the frame log shows animation details like frame number, Key Frames, length of motion paths etc. The *Shifting Keys* and *Scrolling Keys* expand the frame settings by the factors inserted and are best left alone to start with. OK all this and you'll see the screen now contains some sort of theatre set up. Make sure you're looking via the camera's view by selecting *Camera* in the *View* panel. You can see certain objects as well as a *ZY* reference grid plane. *X* is left and right, *Y* is up and down and *Z* is in and out.

Click on *Preview; Wireframe; OK*. The scene will be rendered in wireframe mode. It will take a few minutes. The resulting wireframe animation can be viewed using the *Play* button with its associated control buttons. Brilliant! Renders can be made from any viewpoint, so you can have a *Perspective* view, or a view from a *Light's* position etc. Obviously, only the *Camera view* will allow panning, zooming etc. The *Camera* button provides an interface to control all this. After you've enjoyed the wireframe animation, free up the RAM afterwards by selecting the *Free Preview* option.
You can stop rendering at any time by pressing the *Escape* key.

You may find that some of the *Scenes* in the *3D* drawer may not be able to locate certain *Object* files directly. You may therefore be asked if you wish to load an alternative *Object*? If you *click 'Yes'* it will search elsewhere and you should be able to locate the file. *All the required files are in there somewhere.*

Colour Rendering

To produce a full colour render (the *Surfaces* such as colours and textures are already in) you must go to the *Record* button first and organise a suitable format and destination for the animation. ***Do this before rendering anything you wish to save!*** *Y*ou can render animations in either 6-bit or 8-bit, subject to memory restrictions. I couldn't do 8-bit Anims with an 8MB SIMM, but a 32MB one gives you it all. Alternatively, you can render individual frames in HAM8 or 24-bit for compilation later, or use 24-bit animation facilities if you have them. Once this is all set, press *Render* and confirm the subsequent requesters.

The render output to the monitor is initially in greyscale, but the cursor arrow will soon change colour and a 6- or 8-bit colour image will start to appear in broad horizontal sections. Left clicking the mouse on the screen changes the the view back back to greyscale, which includes a progress report. Clicking on the screen with the RMB gets back to the HAM display. Alternatively, press *LeftAmiga+M* keys to cycle through the active screens. You can get back to Workbench and launch another programme if you wish. Subject to memory restrictions, LightWave should multi-task with other programmes, but it slows things down of course. Beware, however, if memory is saturated you may crash LightWave and lose the job!

In the *Layout* interface, you can modify the *'Space Fighters' Scene* as much as you like. Using the button set at the top of the screen, almost anything can be achieved. For example, try adding some extra lights via the *Lights* button. Maybe place a red spotlight (pointing outwards!) in the exhaust nozzle of each ship and then *Parent* the light to the ship. It will cause the light to move in precisely the same path as the ship. Try adding some flares by *Enabling Lensflares* from the *Lights* requester. Experiment with all the flare parameters, they are superb! The colour/brightness of the stars can be adjusted via the appropriate *Surfaces* box. *Luminescence* is self-brightness, independent of others lights.

Whenever you move anything to a different position on the *Layout* stage, remember to press the *Create key* button, otherwise it forgets the changes and renders the default or the last saved position. Object movement between Key Frames is interpolated automatically, either in 'straight' lines or according to the schemes controlled by *Splines*.

LightWave remembers any object rotations created with the mouse between Key Frames. You don't have to Key Frame every half rotation to get them to render. You can *Render* as many or as few frames as you want, in either *Automatic* or *Manual* frame advance modes. *Manual* requires a left mouse click between each frame render. Study the *Render* requester carefully to make sure it does exactly what you want it to do.

Your colour-rendered animation can be refined outside LightWave, using something like MainActor Broadcast if you need a sound track or ClariSSA to augment it through interlacing. DPaint might do at a pinch, but I find MainActor excellent. LightWave will generate standard ANIM-5 files, or for better results, individual IFF frames up to 24-bits deep. You need a big drive for these, which can take up 2MB or more per frame. A 24-bit IFF

frame to ANIM-5 converter to try is a PD programme called 'Rend24' by Thomas Krehbiel. I find it works very well and it *was* used by the 'Babylon 5' people so it can't be bad! You can also generate more efficient ANIM-7 files from HAM8 frame renders using MainActor. Play your animations through any of the above packages, through Viewtek or record them on video tape.

Note - Memory

One point to note about LightWave is the way it reacts to lack of memory, which is always likely to happen. The programme is so enthralling, you may try to push it beyond your computer's resources. In such a situation, it may or may not inform you of the fact. For example, when a *Scene* is *Loaded*, the Layout interface requires a memory buffer to accomodate its *Bounding Box* data, etc. You may find that large *Scenes* overload the buffer and you'll get a message to that effect. Sometimes, clicking *OK* will allow things to proceed, omitting the *Bounding Box* display. On other occasions, you may not get any message like '*Not enough memory'*. It may just say '*Can't open Anim-file*' or some such comment, which can leave you slightly bewildered. Sometimes, the programme will crash for no apparent reason. For example, a crash may result if you have insufficient *Segment Memory* allocated in the *Camera* menu. The solution to all these situations is to fit a larger SIMM.

LightWave Tutorials

The following Tutorials are based on practical excercises covering the subject matter discussed in each section. The LightWave graphics are derived from screen grabs made during the writing of the instructions. Hopefully, this will make them very easy to follow and replicate.

Tutorial 1: Loading and Rendering a Scene

The Space Fighters Animation

Let's take a look at the characteristics of a typical LightWave *Scene.* The following tutorial will familiarise you with how to load a *Scene* form the *3D:Scenes* directory and what happens when you do. It will demonstrate how a simple arrangement of *Objects*, *Surfaces*, *Camera*, *Motion Paths* and *Motion Blur* is able to produce stunning results in a very short time.

Open the Layout interface. You see the *Grid* and the rear of the *Camera*. You are observing the Layout stage from the *Perspective* viewpoint. Click on different buttons in *View* group and you will soon get the 'feel' of the stage on which *Scenes* are played out. Note that the default position of the *Camera* has its lens centred on the *XZ* plane, looking in the *+Z* direction. So, when you click on *Camera* in the *View* group, you see a horizontal line across the centre of the window. This is the *XZ* plane as seen by the *Camera*. If you click on the *Light* button in the *View* group, the window is blank. That is what the current (default) *Light* 'sees' because the default *Light* is a distant type. This means it simply faces in one direction, but doesn't actually focus on anything. It shines 'everywhere' in the direction set by its *Rotate* parameters. You can visualise the *Light* by clicking on the *Light* button in the *Edit* group. This makes the *Light* the *Edit* item and therefore selected. When selected, an item is drawn in yellow (highlighted). You can only select one item at a time. If you click on the *Camera* button in the

Edit group, the *Camera* will become highlighted and you will see its field of view as a dotted outline shaped like a square pyramid. Swap from one *View* position to another to see the general shape of the *Camera* 's field of view. LightWave's *Camera* has a rectangular field to make it easier for you to 'fit' it's pictures onto the screen.

To load an existing *Scene*, click on the *Scene* menu button on the top left of the interface. The *Scene Editor* pops up.

Click on the *Load Scene* button and the *Load Scene File* requester pops up. This will search the default path for a *Scene* file i.e. in the *3D:Scenes* directory.

Locate the required *Scene* file, in this case it's called '*SpaceFighters*'. Click on the name to insert it in the *File* field. The recommended suffix for a *Scene* file is '*.scn'*, which immediately tells you what type of file it is. Click *OK* to *Load* the *Scene.*

The different items contained in the *Scene* file will appear in the *Scene Overview* panel, together with other details. You can see that the *Scene* consists of *Objects* (4), *Surfaces* (13), *Lights* (1), *Polygons* (8178) and *Images* (4). The *Surfaces* data and the *Image* files are extracted from the *3D:Surfaces* and *3D:Images* directories automatically. These locations were recorded within the *Scene* file when it was *Saved*. Some *Scenes* may need to search other sources and a message will pop up if the relevant directory cannot be located.

The *First* and *Last Frame* numbers and the *Frame* Step are also displayed. The latter figures tell you that the rendering process will deal with all *Frames* from 1 to 200. This *Scene* can obviously provide up to two-hundred separate *Images*, suitable for generating an animation file. A *Scene* with only one *Frame* can only produce a single *Image*, though a 'blank' *Scene Editor* defaults to thirty *Frames*. If you need more or less *Frames*, you simply type in the appropriate *Last Frame* number and Layout will accommodate you.

Each item in the *SpaceFighters Scene* is listed in the *Overview* se
ction of the *Editor*. You will notice the icons representing the four *Objects* ( ), the *Light*
( ) and the Cam
era ( ). The next column contains their visibility indicators. The four *Objects* are set to *Full Wireframe* visibility ( ), the *Camera* will be visible
( ) but the *Light* will not be visible. It
s indicator field is blank. If you click the mouse cursor in the vacant field, the *Light* visibility will be switched 'on' ( ). You can repeat click in any of the other indicator fields to amend their visibility in the Layout window. Ensure they are all at full visibility before moving to the next step.

To the right of the *Scene Overview* panel is a *Frame* chart containing dotted white lines in the rows opposite four items. These indicate that there are several *Key Frames* associated with them. Each of their *Key Frames* is marked with a little cross at certain *Frame* numbers. You can scroll the *Frame* chart using the slider button at the bottom. The other items (the *Light* and the *Random Stars Object*) have no line, but instead have a single cross at *Frame* 0. This indicates that *Frame* 0 sets all their positional and rotational parameters for the rest of the *Scene*. In other words, neither item moves or rotates at any time throughout the *Scene*. Note that *Frame* 0 is always a *Key* for all items, but it need not be the first *Frame* in the rendering schedule. LightWave defaults to *Frame* 1 for the rendering stage, but you can change this if you wish. Click on the *Continue* button and you will be returned to Layout.

The window will display several items in the *Scene* depending upon which *View* you have selected. Click on the *Camera* button in the *View* group to see what the *Camera* sees. The *Grid* has moved into an oblique angle. This is due to a shift in the *Camera* position, because the *Grid* is fixed in space. You can see a 'background' of points. These are all *Single Point Polygons.* They are grouped together in their thousands (5,094 actually) to form a large spherical cloud. This is the *Random Stars Object* seen in the listing above. If you *Load* the *Random Stars Object* into Modeler, the grid squares need to be at about 1 kilometre in size to see anything at all! At 20 kilometres per *Grid* square, the *Object* just fits the Modeler windows.

A *Space Fighter Wireframe* is also visible in silhouette. There are two such machines in the *Scene*, so to determine which this is, click the *Objects* button under the *Edit* group. The currently selected *Object* will be drawn in yellow. It could be any of the four *Objects* and it may not even be in sight at this stage. The selected Item scroll bar beneath the window contains the name of the highlighted *Object*. If the *Space Fighter* we want is not highlighted, click onto the scroll bar and move the cursor over *SpaceFighter (1)* or *SpaceFighter(2)* until the one in view is selected. It's actually *SpaceFighter (1).*

With *SpaceFighter (1)* selected, not only is it drawn in yellow, you can see a white *Motion Path* extending before it. This contains a white cross, just as we saw in the *Overview* chart It represents the location in space of a *Key Frame* for *SpaceFighter (1).* If you look closely at the *Motion Path*, you will see it has dots all along its length. These dots are *Frame* markers. This is where the *SpaceFighter (1) Object* will be located at each *Frame* in the animation. The *Motion Path* is visible in the Layout window because the *Show Motion Paths* button in the

Options menu is switched on (selected). You can change this at any time by clicking the *Options* button and amending the *Options Editor* as necessary.

The *Camera View* is currently that for Frame 0, as indicated by the *Current Frame* number just below the scroll bar. To see the *Camera View* at any other *Frame* number, just type the required number into the *Current Frame* field, or use the *Frame Step* buttons (**<**) and (**>**), or the *Slider* button. However you alter the *Current Frame* setting, every *Object* is replaced by a *Bounding Box* for the purposes of computation. When LightWave has computed the new *View*, the *Wireframe* images will be displayed after being *Redrawn* into RAM. You can watch this *Redraw* occurring in real time by clicking on the *Show Redraw* button in the *Options* menu. It really slows things down, so don't bother. As you step through the *Frame* numbers, you will see that movement has been assigned not only to the *Space Fighters*, but also to the *Camera* (you will notice that the orientation of the *Grid* is changing). To really appreciate the action in this *Scene* we need to run a *Preview* animation.

Click and hold on the *Preview* button at lower left of the Layout display. Move the cursor over *Make Preview* and release the LMB. The *Make Preview* control panel pops up. This will default to the *First Frame*, *Last Frame* and *Frame Step* settings we saw on the *Scene Overview* panel. You can select the format for the *Preview* as *Bounding Box* or *Wireframe*. Ensure *Wireframe* is selected and Click *OK.* The *Preview* animation will be computed, while the cursor shows *'busy'*.

Each *Frame* of the *Preview* will be computed and displayed as it is drawn in RAM. *Previews* are drawn in black throughout, including the *Motion Path* of the last selected *Object*. The *Frame* number is displayed in the lower right corner of the window. After a couple of minutes, the *Preview* will be ready and you will be returned to the starting position (*Frame* 1). The Layout display will be ghosted except for the *Preview Playback Controls*. The *Preview* is currently paused, as indicated by the selected *Pause* button. The default playback speed (*Frame Rate*) is 25 frames per second (fps). You can change this if you wish. The default provides a very smooth *Preview*.

Click on the *Play Forward* button (**>**) to start the *Preview* animation. You will see all two-hundred *Wireframes* animated at 25 fps and then loop back to the start. This will continue until you *Pause* the playback or press another control button. You can play it forwards or backwards, step *Frames* or go to the start or end *Frame*. When you have finished the *Preview*, click the *End Preview* button to return to the Layout display. The *Preview* file will stay in RAM until you select the *Free Preview* option on the *Preview* scroll bar. If you wish, you can *Save* the *Preview* animation by returning to the *Scene Editor*. At this point, the *Scene* exists in the form of the *Preview*, so click the *Save Scene* button to pop up the Save Scene File requester. This will default to the *3D:Scenes* directory. However, you should *Save* the *Preview* to the *3D:Previews* directory. Change the path to *3D:Previews* and *OK* to *Save* the *SpaceFighters Preview*. Later, this file could be loaded into Layout as an *Image Sequence* and used for various purposes, such as a *Background Sequence*. This facility is accessed via the *Images Editor* (to load the *Image Sequence*) and the *Options Editor* (to use it as a *Background Sequence*). If RAM is limited, select the *Free Preview* option under the *Preview* button to clear the animation from memory.

Back in Layout, you can explore the menu buttons to get various information about the *SpaceFighters Scene*. The *Objects Editor* will provide details of each *Object*, including their *Point* and *Polygon* counts. You can control whether an *Object* will, or will not, cast or receive shadows and a lot more besides. The Surfaces Editor will tell you about all the *Surface* colours and other physical characteristics assigned to named *Surfaces* within the *Objects* contained in the *Scene*. The *Images Editor* will show thumbnails of all the *Image* files being used in the *Scene*. In this case, you will find several *Images*, or *Maps*, which have been assigned to the various sections of the *SpaceFighter Object.* Each *Image* map is given a specific name so that its use on the *Object* is logical. The *FinMap Image* is *Texture Mapped* onto the *Fin Surface* as a *Planar Image Map.* The other main sections are handled in a similar fashion. These maps have been designed using a paint package and the saved as brushes. They should really match the shape of the *Surface* they will be projected on and also have a black background. This is where a screen grabber can be handy for collecting the shapes of *Object* parts in Modeler and transferring them to DPaint or whatever for colour finishing. Simpler parts of the *SpaceFighter* are coloured and given reflectivity, etc. directly via the *Surfaces Editor.*

To produce a full colour animation from the *SpaceFighters Scene*, click the *Record* button. This pops up the *Record Editor* with which you can manage all aspects of the process. You may wish to save each of the two-hundred images as a separate 24-bit file. This can be set using the *Save RGB Images* button. You can also watch the rendering process by assigning a 6 or 8-Bit (or other) *Render Display*. Let's save the animation as an ANIM file. LightWave v3.5 produces these in ANIM-5 format. If you wish to generate a different format you could use an external converter, or compile the ANIM-file from 24-Bit Frames, etc. All this is done outside LightWave.

Click on *Save ANIM File* to pop up the *Save ANIM File* requester. This will default to the Video Toaster's non-existent *Framestore* directory, so change the path to your preferred location. Call the file *'SpaceFighters'* or something appropriate. Click *OK* to confirm the *Save*, then click *Continue* to return to Layout. Here, click the *Render* button in the bottom left corner of the Layout display. This pops up the *Render Scene* panel. This will confirm details of the operation, the *First Frame*, *Last Frame* and the *Frame Step*. To permit unattended *Frame*

rendering, click the *Automatic* button, otherwise you will have to activate *Next Frame* render using the LMB. Click *OK* to initiate the rendering.

The screen will switch to the render display. This will show a monochrome render then switch to the selected *Render Display* format when the *Frame* is fully integrated. Each *Frame* rendered will update the display. If you click the LMB at this stage, you will be taken to the monochrome render progress screen. This displays various information about the process, such as the number of the last frame rendered, the render time, etc. You can get back to the colour render display by pressing the left *Amiga/M* key combination. After a period of rendering, the display will return to Layout. To play the animation, you must load it into a suitable ANIM player.

When you play the animation, note how *Motion Blur* has been used to smooth the background stars into streaks. Without this setting, the *Single Point Polygons* would stutter across the screen in an unsatisfactory way.

This *Scene* might be refined by inserting red *Spotlight* in the exhaust nozzle of each *SpaceFighter.* Then using the *Parent* method, you can ensure that the exhaust glow follows the *Fighter*.

Tutorial 2: Maximum Smoothing Angle - Theory

The Smoothing Threshold© Concept

The *Maximum Smoothing Angle* (*MSA*) is of fundamental importance in getting LightWave *Objects* to render and animate well, particularly when *Bones* are involved. The *MSA* appears as a variable data field in several menus and unless an appropriate figure is inserted, strange and unsatisfactory results can be obtained. Unfortunately, all explanations of the MSA which I have read in LightWave manuals and in articles written by LightWave professionals, are confusing. This is not helped when one finds quite contradictory information given in NewTek's own documentation. After looking at the topic from a practical perspective, I believe a simpler way of dealing with the MSA has been found. I've call it the *Smoothing Threshold©.*

The *MSA* affects the ability of the *Smoothing* algorithm to build a smoother boundary between connected *Polygons*. The mechanism only works on *Polygons* sharing a common edge and two *Points* along that edge. *Smoothing* also affects the ability of Layout's *Surface* functions to render a smooth looking boundary that would otherwise appear angular. This ability is known as *Phong Shading*. The *MSA* in Modeler pops up within the *Subdiv* (*Subdivide*)/*Smooth* command in the *Polygon* menu. When you wish to apply the *Subdivide Smooth* command, you are given the option of using a more appropriate *MSA* than the 89.5° default, assuming you know what a more appropriate value would be! This is where a clear understanding of the *MSA* concept is needed. Unfortunately, the conventional approach requires a degree of lateral thinking and can make an interesting Modeler task into a chore.

The 'official' definition of the *MSA* describes how it , '*...determines the angle beyond which LightWave cannot smooth over the seam between adjoining Polygons*.' So, the *MSA* is the maximum angle between adjoining *Polygons* that can be smoothed. Well no, actually, it's not. That interpretation suggests only very angular seams can be smoothed, when the opposite is the case. So, understanding how it, 'd*etermines the angle*', is the clue to understanding the *MSA*. A more accurate definition of the *MSA* is:

> *The maximum angle subtended by the projected Normals of adjoining Polygons, up to which the seam will be smoothed.*

So, if the *Normals* meet the criterion, then the *Polygons* will be *Smoothed*. Confusing isn't it! The best thing here is a diagram. The following shows two pairs of adjoining *Polygons*, seen 'edge on' so their seam angle and their *Normals* can be clearly visualised. The *Polygons* may be *Single-* or *Double-sided*, the principle is unaffected.

In this diagram, the first pair of *Polygons* (P1 and P2) are inclined at an acute angle (A) to each other. Conversely, *Polygons* P3 and P4 are obtuse (B). This arrangement has been made to illustrate the relevance of the default *MSA* setting of 89.5°. An important point to note about these diagrams is that angles A and B are related to SA1 and SA2 as follows:

A = 180° - SA1
B = 180° - SA2

The proof of this relationship is a simple exercise in trigonometry and will not be considered further here. The rule to remember is that the opposite angles of any quadrilateral add up to 180°. So, what we need to determine is which seam will be smoothed under the default setting.
In the first pair of *Polygons*, SA1 is obtuse and therefore larger than the default setting of 89.5°. Accordingly, S1 *will not* be smoothed. Remember, the *MSA* is the *Maximum* angle that will be smoothed. In the second case, N3 and N4 subtend an acute angle, therefore S2 *will* be smoothed. It becomes clear that the default *MSA* of 89.5° is to allow any *Polygon* inclination which is *just* obtuse (at least 0.5° so) to be smoothed over by the *Subdivide/Smooth* command. It will also be rendered smooth by *Phong Shading*. Conversely, acutely inclined *Polygons* are not smoothed at the default setting.

This explanation of the *MSA* is fine in theory, but what setting should one use in practice? It's rather difficult to visualise intersecting *Normals* in a complex *Object* mesh! This is where the *Smoothing Threshold* idea can be of practical help when using Modeler. Think of the *Smoothing Threshold*© as the complementary angle to the *MSA*. The *Threshold* is the *Minimum* angle at which smoothing will occur. It's analogous to A and B in the diagrams. Under default conditions, you can see that the *MSA* and the *Smoothing Threshold*© are each *about* 90°. However, when you examine an *Object* closely, you will see it contains *Polygons* at various angles. Some may be larger than the *Threshold* and others will be smaller. Those which are larger will be smoothed out. So if you decided to reduce the *Threshold* considerably, say to 30° , a lot more seams will get smoothed. However, you might find many seams which should be kept sharp will become rounded. Conversely, raising the *Threshold* to say 120° will enforce considerably more restraint in the smoothing process. *Subdividing* without extravagant *Smoothing* is usually the objective of the *Subdiv* command. Unfortunately, there's no such button as the *Threshold Limit* control! So, after assessing an appropriate *Threshold* from the general character of the *Object*, you have to express it in terms of the dreaded *MSA*. It's easy enough. Just subtract your estimated *Threshold* from 180° and use the result in the *MSA* setting.

The following Tutorials will show you how the *Smoothing Threshold©/MSA* setting affects the results you get from the *Subdivide/Smooth* command in Modeler.

Footnote

In mid-2001, NewTek's release of LightWave v6.5 and subsequently v7 for the PC and Mac platforms, included a new approach to the Maximum Smoothing Angle. They've called it the 'Smooth Threshold' (!). Now this may be pure coincidence, but it's inclusion in their first upgrade since the publication of WaveGuide looks suspiciously like plagerism. Having said that, their explanation of the smoothing principle seems no less confused than it did in version 3.5!

Tutorial 3: The Smoothing Threshold© - Practical

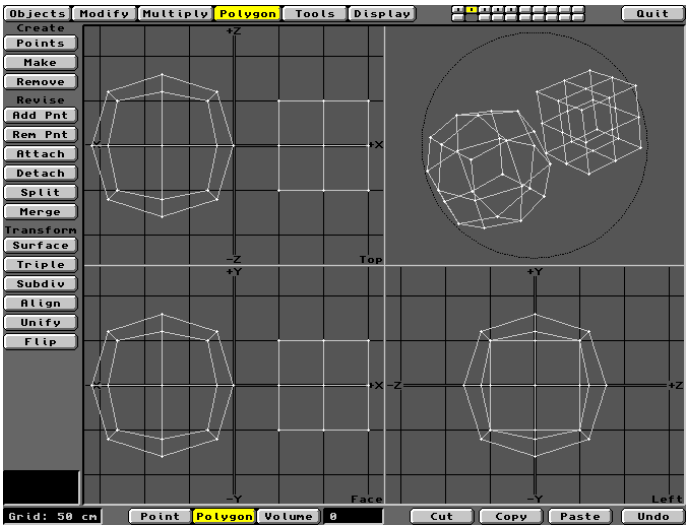
A Ball from a Box

The *Smoothing Threshold* principle is most easily understood by using *Subdivide/Smooth* on a cube. The edges of a cube have adjoining *Polygons* at 90°. So, let's examine a *Threshold* value of 88° and a value of 93°. Using 88° ensures any edge *Polygons* inclined at 88° or more will be smoothed. This means all edges of a cube should be *Smoothed* as we *Subdivide* them. A *Threshold* of 93° , however, prevents any smoothing of the

edges, no matter how many iterations we apply. That's pretty clear, so let's see what happens in practice.

The starting Cubes

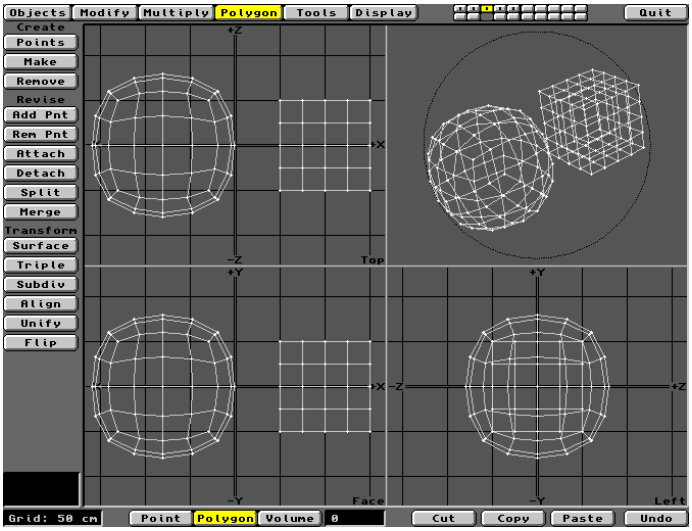
The first graphic shows Modeler with two cubes, each constructed from six square *Polygons*. The cube on the left in the *Top* and *Face* views will have the *Smoothing Threshold* of 88º. That on the right it will have the *Threshold* of 93º. Translating these into the mysterious *Maximum Smoothing Aangle* (*MSA*) gives angles of 92º (180º - 88º) and 87º (180º - 93º) respectively. These are the settings we have to use in the *Subdivide/Smooth* commands. The next graphic shows our cubes after one application of *Subdivide/Smooth* using the above *MSA*s.



The first Iteration of Sudivide/Smooth

We can immediately see the effect of a less-than-90º *Threshold* on the first cube, which Modeler is beginning to *Smooth*. The right hand cube, with a more-than-90º *Threshold* , remains a cube even though its *Polygons* have been *Subdivided* with *Smoothing*. The *Smoothing* is simply not applied. A second iteration of the command

gives us the following.



The second iteration of Subdivide/Smooth

Here, we see yet more *Smoothing* of the left hand *Object*, which has now lost its shape and is tending towards a spheroid. On the right, the cube remains intact. Two further iterations provided the next graphic.

The fourth iteration of Subdivide/Smooth

A ball out of a box! This exercise illustrates the importance of the *MSA*. It also illustrates how the *Smoothing Threshold* principle can be used to get the *MSA* setting right. This is vitally important when you want to *Subdivide Polygons* with a *Smoothing* effect, yet not alter the basic shape of the *Object*.

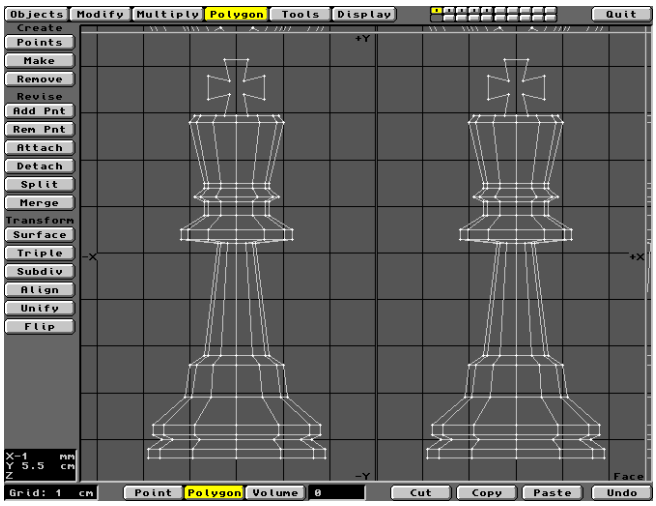
A more serious demonstration of *MSA* and the *Smoothing Threshold* is given in Tutorial 4. This involves the *Subdivide/Smooth* command on a simple chess piece. This will then be animated using LightWave's *Bones* function.

Tutorial 4: Subdivide/Smooth

The Right and Wrong Way to do it

The components of a chess set can be found in the *3D/Objects/Games/ChessPieces* drawer. You can follow this exercise by placing two copies of the '*Light*' or '*Dark*' *King* chess piece in Modeler. When you first *Load* King number one, it will appear as a small spot in the centre of the *Grid*. Since Modeler's default *Grid* setting is 50cm, he looks rather diminutive. To fix this, click on the *Display* button and then click *Fit All*. This zooms your view until the *Object* just fits all the windows. The *Grid* size is now 2cm. You can get the same result by repeat clicking the *In* button. Don't *Load* the second King just yet.

The King chess piece is constructed and saved in unconnected sections. Therefore, if you select any single *Polygon* with the mouse and then press the ']' key, all connected *Polygons* in that section will become selected. In this instance, use of the ']' key to select connected *Polygons* will not select *all* the *Object*. You could use the *Lasso* method to select everything, but the best approach is to join all the sections together at the onset. This is easily done through the *Merge Points* command in the *Tools* menu. The sections are already assembled in their proper locations with a large number of superimposed *Points*. These simply need *Merging*. Click on the *Tools* menu and then on *Merge* in the *Points* group. Select the *Automatic* option and some seventy-two *Points* will disappear as the sections are fused. Note that the cross on top of the crown will remain a separate bit because it had no superimposed *Points*. Deselect everything and then use the *Move* command in the *Modify* menu to shuffle King number one over to the left in the *Face* view. *Load* another copy of the King *Object* and repeat the *Merge Points* command to get King number two prepared. Shuffle him to the right with a *Lasso* selection (RMB) so you can include his cross and then use *Modify/Move*. The Kings are now ready for treatment. Drag the *Face* window to full screen. You can include the crosses in the exercise if you wish. Your Layout display should look like the following graphic.



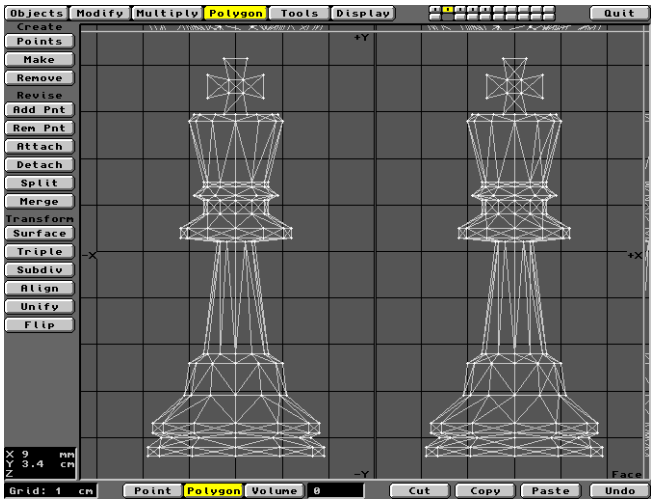
Face View of two Kings ready for treatment

The King's body was generated by *Latheing* a set of profiles into twelve segments. It's not circular the in *Top* view and not very suitable for close-ups because the *Polygons* are too large. Let's assume we want to make an animation in which a King will bend over as if bowing. Clearly, he's not in an ideal state for any *Polygon* deformation routine. Most of his *Polygons* are large and quadrangular, which will neither deform nor render properly. Therefore, we need prepare the *Object* a little before it can receive a *Skeleton* of *Bones*. Exactly how we go about this can mean the difference between success and failure. The left King will receive the wrong treatment. The right will be, well, right.

Look carefully at the *Object*, noting where flat surfaces should remain flat, however much we *Smooth* him down. The underside of the base is a good example. Whatever happens, this should remain flat and at right angles to the bottom ring of the base Also, the top of the 'crown' on which the cross sits is flat. We should try to maintain this shape whilst increasing the *Polygon* distribution to facilitate the use of *Bones*.

The intended use of *Object Skeleton* means you need to work with triangular *Polygons* throughout. The first thing to do therefore, is to invoke the *Polygon/Triple* command with both Kings selected. You can do this without formally selecting anything, because we want all the *Polygons* of both *Objects* to be *Tripled*. Later, we'll need to select one or other King to apply the different *MSA*s. Before doing so, however, it is always worth checking that the previous treatment (*Triple*) has not caused potential rendering errors. It's not likely here, but you must always avoid superimposed *Points*. This is again done through the *Merge* (*Points*) command under the *Tools* menu. As expected, there are no potential problems. However, you should get into the *Merge Points* routine

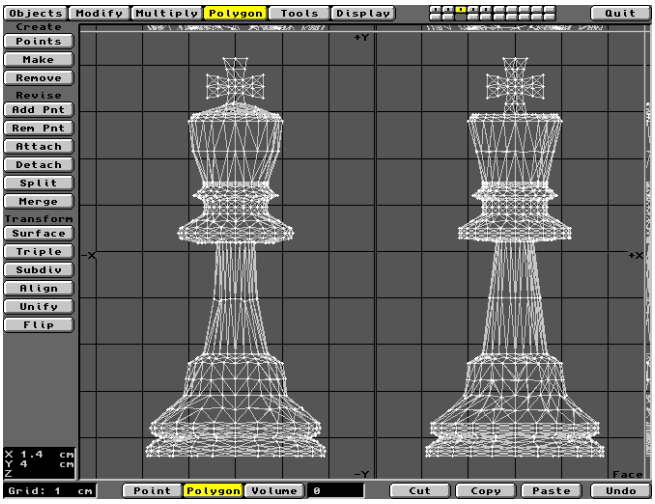
whenever you are modelling. Always check imported *Objects* for superimposed *Points* and save a lot of heartache later. The *Tripled* Kings are shown below.



The Kings following the Triple routine

So far, so good. All the *Polygons* are triangles, so all the routines we apply from hereon will produce smaller triangles. We are now ready to apply the *Subdivide* command to reduce the size and increase the number of triangles. We want the chess piece to become rounder and smoother as we multiply the *Polygon* count. This is where the *Subdiv/Smooth* routine comes into its own. Clicking on the *Subdiv* button in the *Polygon* menu pops up the *Subdivide* control panel. We need the *Smooth* option. This invokes the *Maximum Smoothing Angle* (*MSA*) data field.

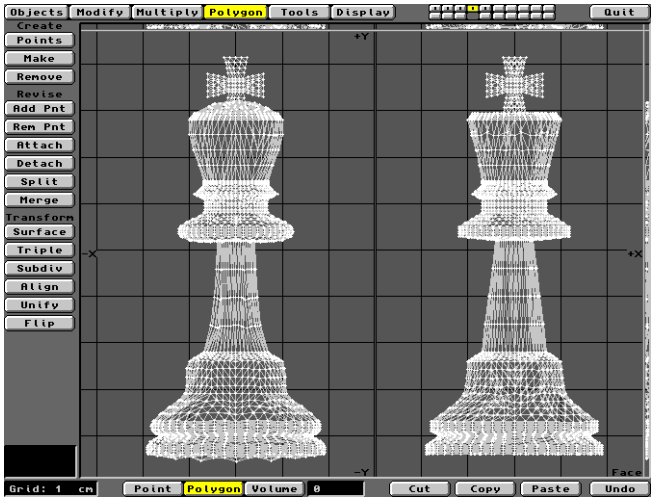
The default is 89.5º which as we've seen, will *Smooth* across any edge *Polygons* inclined at angles of 90.5º and higher. Looking at the chess piece, there are lots of surfaces at angles greater than 90.5º and we really need to keep them in their correct orientation. Remember, we are not trying to radically alter the profile, we are simply trying to increase the number of three-point *Polygons* and smooth the outline where appropriate. However, most of the facets on the King seem to be inclined to each other at about 135º or less. Therefore, this looks like a good starter for the *Smoothing Threshold©*. Above this angle, we could accept some smoothing. Let's insert an *MSA* of 45º for the 'right' King. This is (180º - 135º) as discussed earlier. For the 'wrong' King, let's keep the default *MSA* of 89.5º, giving a *Threshold* of 90.5º. This means any *Polygons* at angles of 90.5º or higher will be *Smoothed*. Selecting each piece individually and applying *Subdivide/Smooth* under these *MSA*s will give the following results.



The first iteration of Subdivide/Smooth

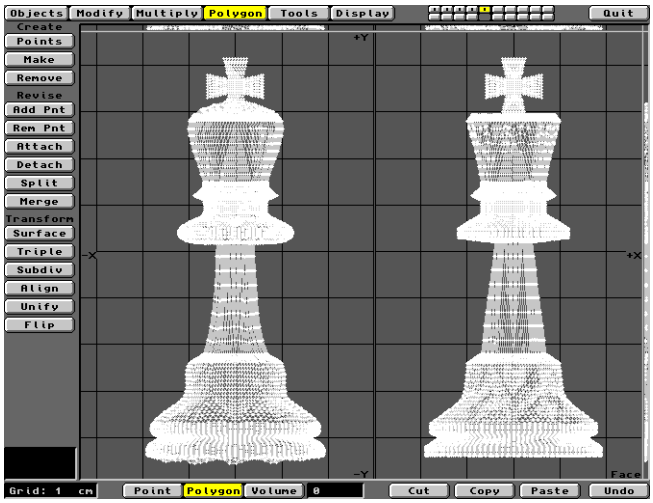
Already there are significant differences. The 'wrong' King now has a domed crown, with some curvature to the sides. Also, the base area has been given curves where we don't really want them and are causing it to lose definition. The stem is starting to curve because its long *Polygons* are inclined at about 120º to the taper on the

base. That's more than the 90.5° *Threshold*, so Modeler is trying to smooth across them. The 'right' King however, has retained his original crispness. His crown is still flat and the base area looks good. All his right angles remain right angles. As before, you should now eliminate any spurious *Points* using the *Tools/Merge* command. Let's give the *Subdiv/Smooth* another go to generate a really large number of triangles. The *Bones* based animation will be all the better for it, though it will take longer to render. Following the *Merge Points* operation, the next iteration of the *Subdivide/Smooth* command will generate the results shown below.



The second iteration of Subdivide/Smooth

The 'wrong' King is really losing shape now, with curves appearing on almost every surface. His crown is ballooning and even the underside of the base is beginning to distort. This is really as far as *Subdivide/Smooth* needs to go for a *Bones* operation and this King is not the man he was. The 'right' King however, retains all the crispness of profile that his designer gave him. This *Object* looks the part and will animate superbly with *Bones*. As a final insult to the wrong King, a third (and *very* lengthy) iteration of *Subivide/Smooth* produce the flabby distortion shown below. On the other hand the right King, all 31,616 triangles of him, remains (almost) perfect!



The third iteration of Subdivide/Smooth

This concludes the tutorial on the *Maximum Smoothing Angle* as utilised by the *Subdivide/Smooth* command. Try to implement the *MSA* using the complementary *Smoothing Threshold*© principle and you should find the routine easier to understand. Of course, the ultimate answer is for NewTek to redesign the *MSA* algorithm. It should really accept a value which LightWave users can relate to by looking at their *Objects* and not the esoteric implementation of *Normals.*

Tutorial 5: Bones

To make the King Bow Down

This exercise will use the 'right' King chess piece generated in Tutorial 4 to demonstrate the power of *Bones*.

Having *Merged Points*, *Tripled* and *Subdivided Smooth* twice, the King chess piece should be *Saved* as a new *Object*. Let's call him "BonyKing". He is now ready to be invested with a *Skeleton* of *Bones.* With this framework in place, he can be deformed in a realistic way. But before we get into the practical aspects, the theory behind *Bones* is worth considering in a little detail. Unfortunately, the subject is given only a cursory glance by NewTek. Even their v5 manual, the latest (and last) of the Amiga manifestations, pays little more attention to *Bones* than did the first.

A *Bone* is a special item which you can *Add* to an *Object* once it is placed on the Layout stage. You do this using the *Objects Editor.* In the middle section of the panel, there's a button labelled *Object Skeleton...*, which gives you access to the *Skeleton for ...* sub-*Editor.* This has a button labelled *Add Bone.* However, the *Bone* is not kept in a directory like an *Object*, it is a special deformation tool which LightWave visualises on the display. A *Bone* is *Added* to the *Current Object*, but it cannot be saved with that *Object*. It is simply a tool which you associate with the *Object* and save as part of the *Scene* file. As with other items, a *Bone* can be assigned *Key Frames* which integrate it into the action. That's why a *Bone* becomes part of the *Scene* file rather than part of the *Object* file. It means the *Object* /*Bone* combination will only appear in that specific *Scene*. If you *Load* the same *Object* into another *Scene*, or *Import* it directly into Modeler, it will not have any *Bones* associated with it.

When first *Added*, all *Bones* are placed at the *Origin* of the *Object* when it was *Saved* in Modeler. It is therefore recommended that you plan the use of *Bones* in advance. It is far easier to manipulate a *Bone* into an *Object* when they are located close together. Therefore, *Objects* should be *Saved* from Modeler with the *Origin* either under or very near the 'base'. This relationship is maintained when an *Object* is *Loaded* into Layout and moved around the stage. The *Object's Origin* is seen as a small '+'. Since the *Bone* is tagged to the Object's *Origin*, things are a lot easier to manipulate. Once positioned, the *Skeleton* will remain correctly orientated with the *Object* wherever it is moved.

When you *Add* a *Bone*, it will be drawn as a stretched-out diamond shape. It will also have a default *Length* of one unit (metre), so it may seem considerably different in scale from its related *Object*. If you look at the *Bone* more closely, you'll see it has a three dimensional structure resembling a spear or arrow head. It has an anchor to the *Object Origin* located at the tip of the 'blunt' end. The pointed shape of the *Bone* is actually irrelevant to its ability to deform the mesh. It's simply a way to help you orientate it in space. It allows you to keep track of its position and direction from one *Key Frame* to the next. It doesn't have a 'front' or 'back' as such. Also, a *Bone's* 'force field' or *Range of Influence* is equally distributed around it. Imagine it like a grossly inflated sausage skin with the *Bone* at the centre. The *Bone's Range of Influence* is very extensive, like the magnetic field of a bar magnet. Thus, a single *Bone* placed in an *Object* will have total influence over it, unless you reduce its influence. This can be done by reducing its *Length* and/or *Limiting* its *Range of Influence*. Naturally, small *Bones* have a smaller influence than large *Bones*, but their *Range* can be reduced without necessarily making them shorter. A *Bone* half the length of the *Object* will influence about half the *Object's* mesh. A full size *Bone* with a *Limited Range* of 0.5 has a similar effect. This dual approach may seem unnecessary, but you will find cases where one or other strategy is preferred. *Bones* also 'compete' with each other, so two similar *Bones* placed in the right position may cancel each other out. To manipulate a *Bone*, you can select it like any other item, so its position and orientation can be edited. You employ the same commands used to *Move* and *Rotate* other items. Its *Length* and its *Range*, however, need special attention, as you'll soon discover.

A *Bone* can exist in three states, *Inactive*, *Active* and at *Rest*. When it's *Inactive*, it has no influence over the *Object*. You can *Move* it and *Rotate* it, with no effect on the *Object* what-so-ever. When it's *Active*, however, any change in its position or orientation will impart similar changes to the *Object*. The *Rest* state is a special case of *Active*. When a *Bone* is at *Rest*, its deforming power is set to a state of equilibrium with the *Object*. It does not deform the *Object* in the *Rest* position, but it is *Active* and 'ready to go'. It's at the starting point from which it 'dead reckons' its location/orientation as the *Scene* progresses.

It's vitally important then, to make all preliminary adjustments to the *Bone*, like setting its location and orientation within the *Object*, while it is *Inactive*. It's in that state when you first *Add* it and on the Layout display, an *Inactive Bone* is drawn with a dotted line. When it's *Active*, a *Bone* is drawn in full line. You can activate a selected *Bone* by clicking the *Bone Active* button on the *Objects Editor* / *Object Skeleton* sub-*Editor*. The keyboard shortcut is 'r', but this should only be used after you have set up the *Rest* state (see below).

To set the equilibrium for the *Rest* state, you must set the *Rest Position*, *Rest Direction* and *Rest Length* using the *Inactive Bone*. To do this, select the *Inactive Bone* and use the *Move* and *Rotate* commands, repositioning it within the *Object* using the mouse. You can change its physical *Length* by clicking the *Rest Length* button. *DO NOT* use the *Size* button for this purpose or things will go wrong! With *Rest Length* on, use the mouse to alter the size of the *Bone* as appropriate. Only the *Length* is important. The apparent width and breadth adjust automatically. (If you wish, you *can* alter these using *Stretch).* When all the settings are correct, press the 'r'

key. This establishes the *Rest* conditions and makes *all Bones Active*. Any movement of a *Bone* will now be followed by the *Object's* mesh.

Of course, a single *Bone* is unlikely to provide the *Object* with the flexibility needed to produce life-like deformations. You therefore need to *Add* further *Bones* in the way described. Once several *Bones* have been allocated, you will find it best to give each one an appropriate name. The default name is *'Bone'*. If several are present they will be numbered *Bone(1)*, *Bone(2)*, *Bone(3)*, etc. in the order they were added. A good strategy is to name them according to the part of the *Object* they affect. Do this using the *Objects Editor / Object Skeleton* sub-*Editor*, pressing the *Rename Bone* button.

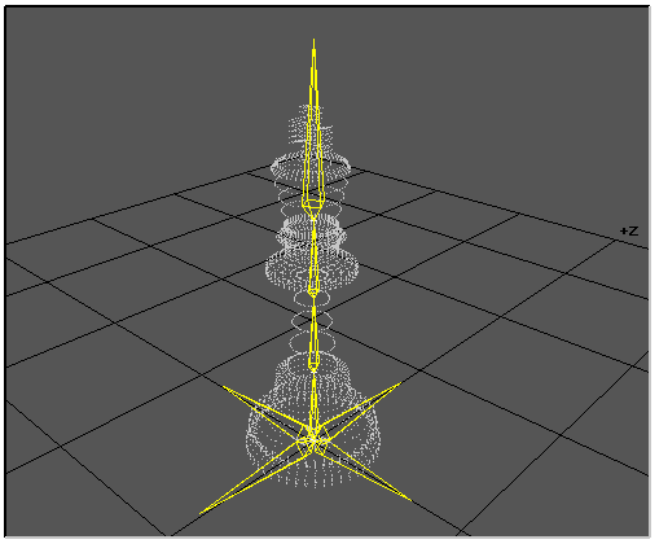
Bones added via the *Add Bone* button have an existence which is independent of the others. They will 'compete' for control of the mesh as discussed, but you can *Move*, *Rotate* and alter their *Rest Length* individually. However, there will be situations where you want several *Bones* to behave in a *Parented* fashion. A good example is where undulating deformations are required, where the movement of one *Bone* has a direct effect on another. This is analogous to a chain of *Bones* like a backbone. To create a backbone, you click on the *Add Child Bone* button. Each click *Adds* a copy of the previous *Bone* in a chain-like manner, with each *Child* attached by its anchor to the 'sharp' end of its *Parent*. These *Child Bones* behave like their *Parent* when edited. If you edit the first *Parent* in the chain, the whole chain responds in the same way. If you edit a *Parent* half way down the chain, only the *Child Bones* beyond it will respond. If necessary each *Bone* in the chain can be individually named as described above.

Building up *Bones* in the ways described will create an *Object Skeleton*, which will give you infinite control of the mesh. The *Rest* state of the *Object Skeleton* should be *Key Framed* prior to the start of any *Bone* activity. This might be *Frame* 0 or a later *Frame* number. In any event, always establish this by pressing the *Create Key* button with the *Bone* selected or use the *All Items* option in the *Create Key Frame* pop up panel. Now let's get back to BonyKing.

In Layout, click the *Objects* button to pop up the *Objects Editor*. Click *Load Object* and select the *BonyKing Object* file from the directory you *Saved* him to. This was logically the *3D:Objects/Games/ChessPieces* directory. When you have *Loaded* the BoneyKing *Object*, leave him at the *Origin* of the Layout *Grid*. Now position the *Camera* to his upper right, so you have a nice *View* looking downwards at about 20°. Note that the *Grid* setting has been automatically adjusted to the scale of the BonyKing *Object*. Each square is now 0.05 (metres), i.e. 5 centimetres across.

Click on the *Scene* button to pop up the *Scene Editor*. Here, change the *Last Frame* number from the default 30 *Frames*, to 50, giving us plenty of *Frames* to make the animation. Now go to the *Scene Overview* and in the second column adjacent to BonyKing, click the visibility indicator to show *Points* only. This looks like nine dots in a block. Click *Continue* to return to Layout.

The King is now displayed in *Points* only. You can easily recognise the shape of the chess piece, but the absence of all the *Polygons* will make the *Bones* easier to see and manipulate. Now click on the *Objects* button to get the *Objects Editor.* In the middle section you'll see the button named *Object Skeleton...* Click on this to get the *Skeleton for "BonyKing"* sub-*Editor* panel. The first button to click here is *Add Bone*. When you do so, the



Current Bone scroll bar will show the name '*Bone*'. The *Bone Rest Length* field shows 1.0 and the *Influence*

Range is also 1.0. These are the default settings for any *Added Bone*. Click *Continue* to return to the *Objects Editor*, then click *Continue* to return to Layout.

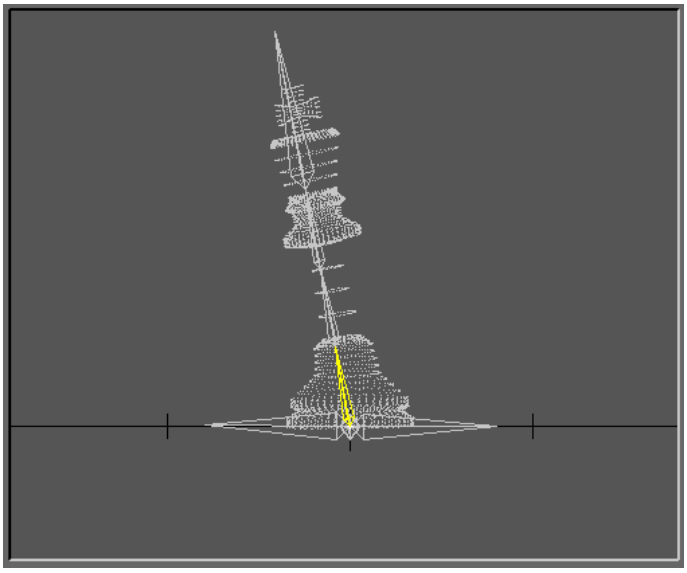
The *Camera View* will now show the King (*Points*), plus a pyramid-like structure with one corner at the *Origin*. This is the *Bone,* displayed at the default length of 1.0 unit (metre). To see it better, take a look from the *XZ View*, though you will have to zoom out some way using the *Edit/View/Zoom Factor* controls. You can zoom with the mouse or type a smaller number into the *Numeric Input* field. Something like 0.5 will be needed to get out far enough to see the shape of the *Bone*. You will see the *Bone* is shaped like an arrow head with its *Origin* placed at the *Origin* of the King, which is also at the *Origin* of the *Grid*.

Click on *Bone* in the *Edit* group to select the *Bone.* It will now be highlighted in yellow. The line is dotted to show that the Bone is *Inactive*. With a default *Length* of 1 metre, this *Bone* is currently far too large for use the King *Object*, which measures only a few centimetres across. The first thing to do therefore is to alter the *Bone's Length*. Click *Rest Length* in the *Mouse* group. This is the button you must use to change the *Length* of a *Bone*. *DO NOT* use the *Size* button or things will go wrong. Drag the mouse to the left with the LMB down to reduce the *Length* of the *Bone*. You will have to zoom back into the *XZ View* to see how things are going. Get the Layout display down to about 2 x 2 *Grid* squares in size. Reduce the *Length* of the Bone until it's about the same as the diameter of the base of the chess piece. You can see the *Rest Length* in the window at bottom right of the display. Something around 0.040 metres will do.

Looking at the *Bone*, you will notice that it was loaded into Layout pointing towards *+Z* and located in the *XZ* plane. The position of any *Added Bone* relative to the *Object* depends on the location of the *Origin* when the *Object* was *Saved*. The BonyKing chess piece was *Saved* with its *Origin* coincident with the Modeler *Origin* and therefore *Loaded* coincident with the Layout *Origin*. A *Bone* is *Added* in similar fashion and always 'points' towards *+Z.*

Click on the *ZY View* and you'll see the *Bone* pointing to the right in the *XZ* plane, exactly where the base of the King piece is located. Actually, we need a *Bone* at this location to 'anchor' the base of the King piece down. In fact we will need three more similar *Bones* to fix the base. This is so that the other *Bones* we will add do not pull the chess piece away from the *XZ* plane when we move them. Remember that *Bones* compete with each other for control of the *Object's* mesh. So, if you want an *Object* to stay put while you bend or twist it, always place some anchoring *Bones* around the base to hold the *Object* down. This *Bone* is now where it needs to be throughout the animation sequence, so click *Create Key/Frame 0/Selected Item* to fix it. Let's put in three more.

Get back to the *Skeleton for "BonyKing"* panel via the *Objects/Object Skeleton* buttons. Click *Add Bone* again and repeat the resizing operation using *Rest Length*. The second *Bone* will be coincident with the first, so use the *Mouse/Rotate* command to allow you to rotate it to 90° in the *XZ View*. Check via the *XY View* that the *Bone* is still 'horizontal'. When it looks about right, fix it with the *Create Key* routine. Do this two more times, resizing and rotating each new *Bone* until you have four of them, effectively pointing 'N, S, E and W' in the *XZ View.*



Remember to *Create Key* for each one at *Frame* 0. You should also *Rename* them appropriately. Something like *BaseBone 1*, *2*, *3* and *4* will do.

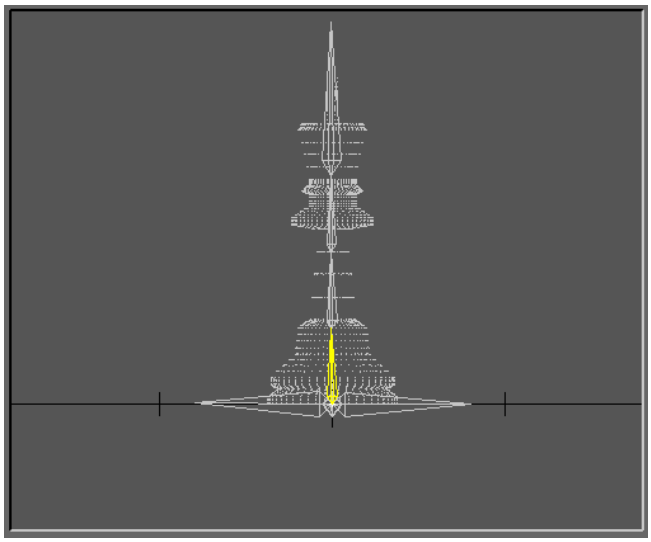
Now, we'll create a 'backbone' of parented *Bones* from the base to the crown. Go back to the *Add Bone* routine, then resize this fifth *Bone* to around a quarter of the length of the *Object*. A *Rest Length* of around 0.02 is near

enough. Using *Mouse/Rotate*, orientate the new *Bone* vertically upwards from the *Origin.* Check it out in each *View* to be sure of its orientation. When you're happy, *Create Key* at *Frame* 0 and *Rename* it *BackBone1* or something similar. Go back to the *Skeleton* panel and click on the *Add Child Bone* button. While you are here, *Rename* the *Child Bone* something like *BackBone2.* Go back to Layout and you'll see that the *Child Bone* is a carbon copy of its *Parent Bone*. It is attached to the 'sharp' end of the *Parent* by its *Origin.* There is no need to resize it . Repeat all this twice more. You should now have a four unit chain of *Parented Bones* extending from the base to the crown of the King piece. To ensure the cross on top of the crown stays put, you could increase the *Rest Length* of the last *Bone* (*BackBone4*) to around 0.04, so that it extends beyond the cross. Remember that a *Bone's* influence is determined by its *Rest Length*, so make sure the cross is within its 'force field'. *Create Key* at *Frame* 0 to fix all this in place. When all the *Bone* manipulations are complete, the best strategy is to use the *All Items* option in the *Create Key* panel to ensure nothing has been missed. Do it now.

So far, all the *Bones* you have *Added* to the *Object* are *Inactive*. You must now equilibrate the *Rest Position*, *Rest Direction* and *Rest Length* of the *Bones* with the *Object*. This tells LightWave that the *Bone* arrangement you have set up is the at *Rest* state. The easiest way to do this is to press the 'r' key as each *Bone* is selected. This confirms the *Rest* parameters and also makes the *Bone Active*. The alternative is to go to the *Skeleton* panel and click the *Bone Active* button for each *Bone* on the *Skeleton*. The *Rest Position* and *Rest Direction* buttons will display the coordinates and orientation of the *Current Bone*. The *Rest Position* of a *Bone* is determined by the coordinates of the anchor point at the 'blunt' end. When a *Bone* is *Active*, its selected image is drawn in full yellow lines. Once *Activated*, any change in any *Bone's* position or orientation will deform the mesh. The *BonyKing's Skeleton* should look something like the following. Note that for the purpose of illustration, this shows all the *Bones* highlighted.

Camera View of BonyKing with his Skeleton of Bones

Switch to the *YZ View* to see the King piece from the side. Let's assume he is 'facing' left, so in our animation that's the direction he's going to bow . We'll make a looping sequence, so he will bow downwards in the first half, then return to his upright stance over the second half of the animation.



The YZ View with BackBone1 selected

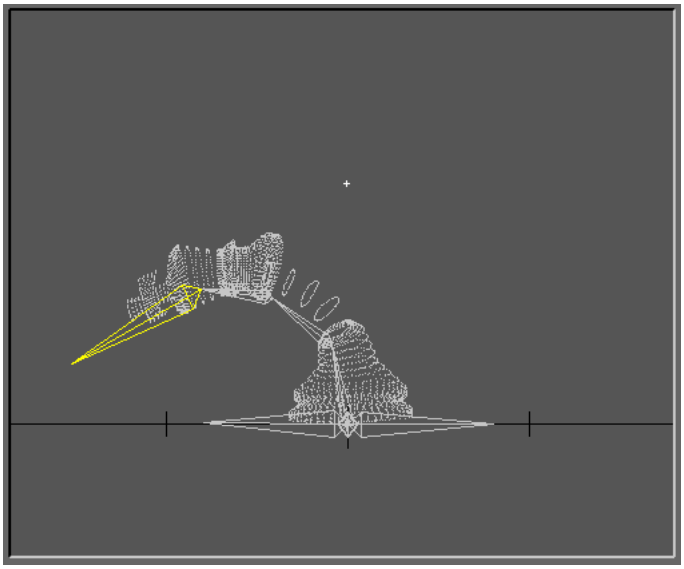
It's interesting to watch *Bones* in action as you edit them. With a simple *Object* like the chess piece displayed in *Points* only, the redraw is virtually instantaneous. Make sure all the *Bones* in the *Skeleton* are *Active* by selecting each in turn and pressing the 'r' key. This is a one-way key. You can only *Inactivate* a *Bone* via the *Skeleton for...* sub-*Editor* using the *Bone Active* toggle.

Press *Edit/Bone* and select the lowest *Bone* in the backbone chain, I've called it *BackBone1* in the graphic. Using the *Mouse/Rotate* commands, rotate *BackBone1* a few degrees anti-clockwise. This is actually a change in *Pitch*. As you rotate *BackBone1*, all the *Child Bones* follow suit. Make sure the chain remains vertical in the *XY View.* The easiest way to avoid the 'wobbles' is to limit the *Rotate* command to *Pitch* by deselecting the *H* and *B* buttons in the *Mouse* group. As you edit the *Bone*, the mesh is deformed around it. The body of the King is starting to tilt because the *Child Bones* have also tilted with the first *Parent*, *BackBone1*. The *YZ View* should look a little like the following screen grab.

Rotate BackBone1 in Pitch

Now select *BackBone2* and repeat the rotation in *Pitch* a little further. Note how *BackBone1* remains put, while the rest of the *Child Bones* follow their next *Parent* in the chain, *BackBone2*.

Repeat this rotation with the next two *Bones* in the chain (*BackBone3* and *BackBone4*) until the King is in the fully 'bowed' position. You should get something like this.



The completed bow using each parented Bone in the BackBone Chain

Note that the base of the chess piece is just about where it started off. There's a very slight tilt due to the large deforming influence of all those *Bones* in the chain. Without the *BaseBones*, however, the King would have almost turned somersault (now there's an idea!).

The King is now in the fully bowed position, so this arrangement should be made *Key Frame 25*. Press the *Create Key* button and change the *Frame* field to 25 using the keyboard. Select *All Items*, because we want all the *Bones*, etc. to be *Keyed* in this condition. Click *OK* (or press *Return*) to confirm the *Key Frame* and return to the Layout display.

The display now shows the *Motion Path* of the *Current Bone*. You can flip through the *Bones* using the *Selected Item* scroll bar and see each *Motion Path* in turn. The *Motion Path* of a *Bone* can be refined if necessary using *Splines*. As with all *Motion Paths*, LightWave applies a default *Spline* effect to all changes in position and direction, so they will appear very realistic without further treatment. However, you may want to amend this into some other *Spline Path*. This can be done using the *Spline Controls* button. This will pop up the *Key Frame Spline Control* panel, discussed elsewhere in the guide.

Go to *Frame* 0 using the *Key Frame/Previous* (**<**) button. This should show the King in the fully upright, starting position. This is because we used *Create Key/All Items* at *Frame* 0 before manipulating the *Bones*. This is also the position we need the King at the end of his bow, i.e. at *Frame* 50. If you click the *Key Frame / Next* (**>**) button twice, you'll flick through *Key Frame* 25 and go back to *Frame* 0. This is because *Frame* 50 has not been made a *Key Frame*. *Frame* 50 needs to be identical to *Frame* 0, so that the King's bow will loop when the animation is played. To ensure this, click *Create Key* when the *Current Frame* is 0. Type 50 into the *Frame* field and select *All Items*. Click *OK* (or *Return* key) to confirm. The condition of *All Items* in *Frame* 0 is copied to *Frame* 50. That completes the preparation for rendering the animation, but the next thing to check out is the *Preview*.

Click on *View/Camera* to see what the *Camera* sees of your work. This is the *View* that LightWave will eventually render. Click on *Preview* to pop up the *Make Preview* panel. This shows the format of the latent *Preview* animation. It should display *First Frame 1*, *Last Frame 50* and *Frame Step 1*. We need to see the animation in *Wireframe*, so ensure this button is highlighted. When everything looks right, click *OK* and let LightWave do its stuff. In a few seconds, the *Preview* animation will be ready to view. Using the *Preview Playback Controls* take a look at the results. You should see the BonyKing making a reasonable attempt at bowing and repeating this loop until you stop the action. *OK* keep it playing; it *is* pleasing to watch your first success with *Bones*!

Tutorial 6: Metamorph

Sqidgy

This is a very simple example of the power of *Metamorph*. It involves the *Morphing* of a sphere into three derivative shapes created with the *Stretch* command.

In Modeler, click *Objects/Ball* and then *Numeric*, to pop up the *Ball* creation panel. Select *Globe* type and then enter a value of 16 for both the *Sides* and *Segments* fields using the keyboard. Make the *X*, *Y* and *Z* radii about 1 metre. Click *OK* to confirm the *Create.* Now click on the *Make* button and the *Ball* will be *Created*. Click on another command button (eg. *Polygon*) to remove the cursor frame. We now have a basic sphere with sixteen sides and sixteen segments. The *Polygons* are mainly quadrangles, which means we could get problems rendering this *Object* in an animation. The safest strategy therefore is to *Triple* all the *Polygons* into triangles.

Select the *Polygon* menu button at the top of the display and then click the *Triple* button in the *Transform* group. All the *Polygons* are now triangular and suitable for deformation, animation and rendering. The Sphere is still a little coarse because the *Polygon* count is still fairly low. It's 480, according to the *Display/Stats* panel invoked under the *Polygon* selection mode (bottom left). Let's increase this to get a really smooth sphere. To do this, you should use the now familiar *Subdiv/Smooth* command in the *Polygon* menu.

Click the *Subdivide/Smooth* command once, leaving the *Maximum Smoothing Angle* at the default setting. We now have a 1,920 *Polygon* sphere, which looks much better. *Save* this *Object* to the *3D:Objects* directory as '*Squidgy1*'. We'll now create a couple of *Morph Targets* from *Squidgy1*. Place two *Copies* of *Squidgy1* in two other *Layers* for use later.

Click on the *Modify* menu button to bring up the command sets. We'll use *Stretch* to create the next *Object*. Click on *Stretch* and place the red *Stretch*-cursor in the *Top Edit Window*. Hold down the LMB and drag the mouse until the sphere has been stretched out along the X axis. If necessary, reposition the window frame to see everything. You should then go to the *Face Edit Window* and use *Stretch* to thin down the *Object* in this view. You should now have a flat elipsoidal shape. Reposition it (*Modify/Move*) so that it's exactly centred with the *Origin* (in all three *Edit Windows*) like *Squidgy1*. If you don't do this, the *Target* in Layout will move sideways to match its new *Origin*. You should *now Save* this as '*Squidgy2*'.

Go to one of the other *Layers* containing the *Copy* of *Sguidy1*. Again use the *Modify/Stretch* command to stretch the sphere in another plane, say along the Z axis. Then, stretch it down to give a flat elipsoidal shape and *Move* it to the *Origin*. *Save* this *Object* as '*Squidgy3*'. Repeat this with the second *Copy* of *Squidgy1*, stretching along the Y axis. Then, after centering it on the *Origin*, Save this *Object* as *Squidgy4*. We now have four *Object*s created from the same shape which we will use in the *Morphs*.

In Layout, click on the *Scene* button to pop up the *Scene Editor*. This is blank at present, except for the *Camera* and the obligatory *Distant Light*. Increase the *Last Frame* number to 100. This will allow you twenty-five *Frames* for each of the four *Morphs*, including one back to the starting *Object*. Close the *Scene Editor* and return to Layout.

Important Note: If you want a series of *Morphs* to loop back to the beginning, you should ensure that the final *Morph Target* is a *Clone* of the *Starting Object.* This is due to the fact that the *Starting Object* is not actually available as a *Morph Target* at the end of the animation. It has itself *Morphed* at that stage. Thus, to get the animation to loop properly, you need two copies of the *Starting Object* in Layout.

Click the *Objects* button to pop up the *Objects Editor*. Click *Load Object* and select *Squidgy1* from the file list in the *Load Object File* requester. Click *OK*. Now click *Clone Object* and enter 1 as the *Number of Clones* required. Click *OK* and the *Objects Editor* will now list *Squidgy1(1)* and *Squidgy1(2)*.

Repeat the *Load Object* routine for *Squidgy2*, *Squidgy3* and *Squidgy4*.

You now have the five *Squidgy Objects* centred on the Layout *Grid*.

Go back to the *Objects Editor* and make *Squidgy1(1)* the *Current Object*. Check that *Squidgy1(1)* has an *Object Dissolve* of 0.0%. This means that *Squidgy1(1)* is 100% visible to the *Camera*.

Make *Squidgy1(2)* the *Current Object* and then make its *Object Dissolve* 100%. This means *Squidgy1(2)* is invisible to the *Camera*. Remember *Squidgy1(2)* is to be a *Morph Target*, so LightWave does not need to have it in *Camera* shot. An alternative way to do this is to leave the *Object Dissolve* setting at the default 0% and simply move all the *Morph Targets* out of sight of the *Camera*. If you do this, be sure to *Key Frame* their locations at *Frame* 0.

Now make *Squidgy2* the *Current Object* and then make its *Object Dissolve* 100%. This means *Squidgy2* is invisible to the *Camera*.
Make *Squidgy3* the *Current Object* and make its *Object Dissolve* 100%.

Now do the same for *Squidgy4* and return to the Layout display.

The display will now show *Squidgy1(1)* alone on the *Grid*. The others are still there but they are invisible. You will see their dotted-line *Bounding Boxes* during the screen redraw, but they will not be displayed. *Objects* which are dissolved have dotted *Bounding Boxes* so you are aware of their presence.

Back in the *Objects Editor*, we now need to set up the four *Morphs*.

Make *Squidgy1(1)* the *Current Object* and make *Squidgy2* its *Metamorph Target* using the scroll bar selector. Now move down to *Metamorph Level* and click the *E* (*Envelope*) button. We are going to *Morph Squidgy1(1)* into *Squidgy2* over twenty-five *Frames*.

The *Envelope Editor* will pop up. This displays a blank *Envelope* graph with *Frame* 0 selected. We need to *Create* a *Key* at *Frame* 25. There are two ways you can do this. Click on the *Create Key* button and enter 25 in the pop up panel or, change the *Mouse Function* to *Create* and simply click on the required coordinate with the LMB.

The *Current Value* should now be set to 100%. You can also do this in two ways. You can type 100% in the *Current Value* field or, change the *Mouse Function* to *Drag* and use the LMB to set the value by dragging on the graph point. (You can alter the position of the *Frame* setting by dragging with the RMB). The typing method is usually more efficient, but it's a matter of personal preference.

You will now have straight line plot between *Keys* (0,0) and (25,100). This line represents the *Morph Envelope* over *Frames* 0 to 25. This would suffice for the *Morph*, but to improve the visual transformation, let's apply a *Spline* control to the event.

With either *Key* selected, click *Spline Controls* to pop up the *Current Key Frame Spline Controls* panel. Enter a value of 1.0 in the *Tension* field and Click *OK*. The *Envelope* graph now shows some 'damping' near the *Key*. Repeat this setting for the other *Key*. You should now have a nice curve between the *Keys*. Click *Use Envelope* to accept the settings. You are returned to the *Objects Editor*.

Make *Squidgy2* the *Current Object* and *Squidgy3* its *Morph Target*. Click the *Metamorph Level E* button and create the *Morph Envelope* for this transformation. Remember this must occur from *Frame* 25 to *Frame* 50. This graph should have *Key* points at (0,0), (25,0) and (50,100). You will notice that LightWave has connected these points using its default *Spline* controls and has drawn a curve through all three locations. When you enter *Spline Tension* settings of 1.0 for *Frames* 25 and 50, everything will sort itself out. Click *Use Envelope* to get back to the *Objects Editor*.

You should now make *Squidgy3* the *Current Object* and *Squidgy4* its *Morph Target*. This *Morph* should occur from *Frame* 50 to 75, i.e. *Keys* (0,0), (50,0) and (75,100) and it should be given the same *Spline Tensions* as before. Click *Use Envelope*.

Next, make *Squidgy1(2)* the *Morph Target* for *Squidgy4*, set the *Envelope* between *Frames* 75 and 100, i.e *Keys* (0,0), (75,0) and (100,100). Apply the same *Spline Tensions* and click *Use Envelope.*

Finally, make *Squidgy1(2)* the *Current Object* and make its *Morph Target* '*(none)*'. This *Object* is the end of the line, but because it's a *Clone* of the *Starting Object*, you will see a smooth loop back to the beginning.

Your *Morphs* are now ready to test, so go back to Layout and position the *Camera* in a location where all the transformations can be observed. We'll use the *Preview* animation facility.

Click *Preview* to pop up the *Make Preview* panel. Ensure the settings are *First Frame 1*, *Last Frame 100*, *Frame Step 1* and *Wireframe* type. Click *OK* and let LightWave calculate the *Morphs*.

When the *Preview Playback Controls* are displayed, click the *Play* control and enjoy!

Tutorial 7: Metaform

The Sports Car in a Shoe Box

Metaform allows you to create realistic 3D objects from simple blocks of polygons. It is particularly effective when a smooth, organic surface is needed. It is important to note that *Metaform* will only operate on three or four-sided polygons. It will subdivide an object and smooth out the polygons in between. It's a bit like sanding down the corners of a block of wood. The degree of smoothing is determined by the angle and distance between adjacent polygons. If you intend to *Metaform* a mesh which contains three-, four- and more-sided polygons, you will be instructed to use the *Triple* command first. Note that *all* selected polygons will be tripled, including the quads. If you want to retain the quads, you should select them separately and hide them *before* using the *Triple* command. This strategy is useful in keeping the polygon count at a reasonable level.

To demonstrate the power of *Metaform* , we'll create the body of a sports car from a simple grid of polygons. This exercise requires the *WaveGuideTutorials* disk.

With LightWave up and running, switch to the Modeler interface and put the *WaveGuideTutorial*s disk in df0:

Click on *Objects* (top left hand button). This prepares the Modeler to do work on a LightWave *Object.*

Click on the *Load* button, situated at the top of the so-called *Fetch* group. A file requester will pop up.

Click on the name of the drive from which the data will be taken, in this case df0: A list of directories appears.

Click on *Tutorial 7.* A list of files appears.

Select *MetaCar.project* and click *OK.*

All ten Layers of Modeler's workspace will be loaded with data. All ten Layer buttons become yellow. They are each marked with a tag (occupied) and activated (upper row). The three *Edit Windows* show all ten Layers at once since they are all selected (yellow).

Click on the first Layer button in the row. Only the data in Layer 1 is now shown. The other Layers become deselected (grey) and now have two buttons. These can now be selected as an upper or lower Layer. Upper Layers may also be referred to as Front Layers. Lower Layers may also be referred to as Back Layers.

Click on the second upper button. Layer 2 now becomes active and replaces the first one. Click along the upper line in turn and see how the different Layers provide different sets of data.

The bottom row of buttons allows any of these other Layers to be placed underneath the active Layer(s). Click any one of the lower buttons. The back Layer will now be seen as a black drawing below the white drawing in the upper Layer.

To demonstrate multi-layer selection, hold down the *Shift* key and click on some other upper or lower layer buttons. Each becomes integrated into the workspace. The active (upper) layers will each respond to design changes concurrently. *Objects* in the back layer(s) remain unaffected.

Go back to Layer 1 with the rest deselected. We will now see how to design a sports car from a simple 3D shape using the *Metaform* function.

Think of a car's shell in its simplest form. It's a just a box for the body with four wheel wells cut out. There's another box on top where the windows and the roof sit. With Modeler, it's even easier. We can start with a 2D profile of some boxes and then extrude them to give some width. This initial 2D profile is shown in Layer 1. The first step therefore, is to make this grid.

To actively construct your own version, Layer 1 will have to be freed up so you can work in it. To do this, press the *Cut* button (bottom right). The grid is removed, though the data is retained in the Amiga's clipboard until it's replaced with something else. You can therefore undo this *Cut* command by pressing the *Undo* button. Try it now.

With Layer 1 empty, we can start on our own version of the grid. You will have noticed that the grid was constructed from quadrangles. Each of the eighteen rectangles was a four-edged *Polygon* (quadrangle). Modeler will automatically construct surfaces in triangles or quadrangles by setting down your preferences in the *Objects/Options* dialogue box.

Press the *Objects* button and then the *Options* command at the bottom of the button bank. The *New Data Options* box pops up. There are various things you can set here. You can pre-name the surface you intend to create, or rename it something else. All un-named surfaces are called *Defaul*t. Press the *Quadrangles* button. This will ensure that all the Polygons we generate are four-sided. The *Automatic* setting will permit *Modeler* to use either triangles or quadrangles depending upon the *Flatness* limit set. A low flatness limit will enforce triangles. The *Polygons* may be single-sided or double-sided. This is relevant to rendering, where *Surfaces* are applied to *Polygons* according to this setting. If a single-sided *Polygon* faces away from the *Camera*, it will appear as a hole. Double-sided *Polygons* are always visible, but take proportionately longer to render.

Press *Two Sides* then *OK*. Making the *Polygons* double-sided in this way allows for easier visibility in the *Preview* window. More on this later. Let's first construct the basic grid from which the car body will evolve.

Click on the *Box* button under the *Objects* menu. The cursor changes to a box. Click on *Numeric* to pop up the *Box* dialogue panel. Here you can pre-set the *Box* parameters. Press *Reset* to bring the settings to default values. The *Low* and *High* columns place the limiting points for the intended box. The *Units* window controls the scale of the settings. We need metres (*m*), the same as the reference grid. Adjust this with the +/- buttons. Let's make the box about 4 metres long and 1 metre high. This will give us a car body about this size in the side view.

To start with, we need the box to have no thickness in the *Top* view and to contain 6x3 quadrangles. So enter the following figures:

	Low	High	Segments
X	-2	2	6
Y	-0.5	0.5	3
Z	0	0	0

Press *OK*. A bounding box appears on the edit screens. Press *Make* and the grid is created.

The next step is to eliminate a few points so that the grid assumes a shape something like that in Layer 2. Click on Layer 2 as a background so you can see the differences.

Click on the *Point* selection button (bottom left). This will allow the mouse to select specific *Points* for cutting out or moving. The upper left point in the Face view for example needs to be cut out. Click on this point to select (highlight) it and then on *Cut* (bottom right). The point is eliminated and the grid closes up with a triangle. Repeat this process where you think a point needs eliminating. The top right point is obvious. Remember there is a single level *Undo* if you make a mistake.

The slope of the car's eventual bonnet can be straightened out by selecting the wayward point, pressing the *Modify* menu button and selecting the *Move* or *Drag* command button. This allows the mouse cursor to be used to move/drag a selected point into another position. Notice that *Drag* will manipulate individual points, whereas *Move* operates on all the selected points concurrently.

To create the front and rear wheel arches requires the removal of two quadrangles. Change the selection mode to *Polygon* (bottom left). This enables the mouse to select *Polygons* rather than individual *Points*. Click the cursor on one edge of the second *Polygon* in on the bottom row. The *Polygon* is selected (highlighted). Notice that all polygons sharing the chosen edge will become selected. You can de-select a highlighted polygon by clicking one or other of its edges. The de-selection routine follows the same rules as selection. You will also notice the selection process draws a surface perpendicular as a broken line. This is called the *Normal* and indicates which direction the *Polygon* is facing. Double-sided *Polygons* have *Normals* on both faces as seen in the *Top* and *Left* views. When only the required polygon is highlighted, click on *Cut* and it will be eliminated, leaving the basis for a wheel arch. Repeat this for the rear wheel arch. The grid should now look something like that in Layer 2.

The next stage involves moving or dragging individual *Points* to create a more streamlined shape to the car's developing profile. Use the *Points* selection button and then the *Modify* menu, followed by *Move* or *Drag*. When there are lots of *Points* to move around, it is best to select all the *Points* in the area and then work on them individually. This grid is a good example. You could select all the *Points* in the grid by clicking each one with the Shift key down. This is OK for selecting a handful of *Points*, but for larger numbers, the best method is to lasso them, as follows.

While in *Points* mode, place the cursor near the outside of the grid and with the RMB, draw a rough circle right around the grid. Release the mouse button. All the *Points* within the line you draw are selected. This works with *Objects* containing many thousands of *Points* and is a great short-cut for selecting them. The lasso doesn't

have to stay within the view screen. Provided the loop is more or less completed, you can lasso everything in a fairly relaxed manner.

Now you have selected all the *Points*, each one can be *Moved* or *Dragged* using the mouse. Just press the *Modify/Move* or *Modify/Drag* command and away you go. Hold down the Left mouse button to drag a *Point* into a new position, then release the button and move to the next *Point*.

You should be able to change the shape of the grid until it resembles that in Layer 3. Note, however, that Layer 3 contains an extra point in the rear bumper area to give it better definition. This is added in *Polygon* mode after first selecting the lower right polygon with the mouse. Now go to the *Polygon* menu (top centre) and click the *AddPoint* button in the *Revise* group. The cursor changes into a *'To'* arrow. The arrow point allows you to place the cursor exactly where on the previously selected polygon you want another point. Click the left mouse button to complete the change. A new point appears in the *Polygon*. Go into *Point* mode and move the new *Point* using *Modify/Move* or *Modify/Drag*. You should now be able to make your grid something like that in Layer 3.

The next step is to extrude this 2D grid into a 3D shape more resembling a car body.

In *Polygon* mode, press the *Multiply* menu button. *Multiply* allows you to increase the number of *Polygons* using a variety of commands, including *Extrude*. We need to extrude the grid along the *Z* axis and to split the extrusion into three sections. This will allow further streamlining of the body in the Top view. To achieve this, press the *Numeric* button, which pops the *Extrude* dialogue panel.

Set the *Segments* at 3 and the extrusion *Axis* as *Z*. The width of the car body is controlled by the *Extent* setting. Set this at 2 metres or thereabouts. Press *OK* and the grid is surrounded by a yellow bounding box containing a T-bar. This T-bar can be used for visual adjustment to the extrusion extent. Press *Make* and the extrusion is made.

The resulting *Object* will have moved away from the *Origin* because we extruded it along the Z axis. The *Object* can be re-centred in the *Top* view using *Modify/Move*, dragging the grid with the LMB. Our car body should now resemble Layer 4. If you'd prefer to view the extrusion as a 3D object, use the *Preview* window as described below.

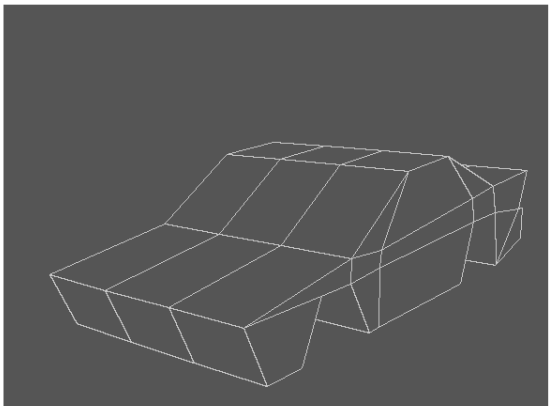
From the *Display* menu, go to *Options*. Under the column headed *Preview*, select either *Static*, to see a static 3D view of the selected polygons, or select *Moving*. There are two additional options within the *Moving* preview. *Wire* gives a moving wireframe view. *Solid* gives a solid representation of the selected polygons. The moving option provides a reciprocating rendition of the polygons. This allows for a clearer picture of how they relate to each other in 3D space. The circle drawn in the *Preview* window acts as a trackball to enable the object to be positioned in an infinite number of positions. Just drag it around with the left or right mouse button.

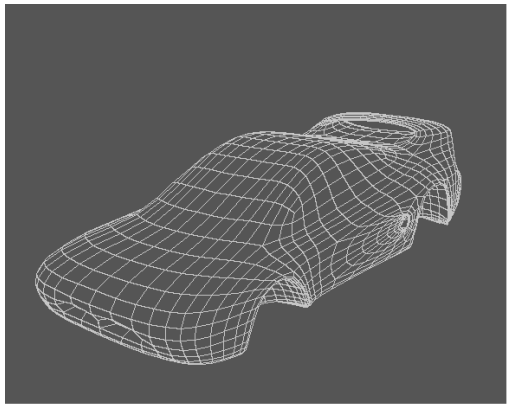
Note that some of the polygons may only be visible from certain angles. This is because they are single-sided. You should see all the polygons in this case, because we selected *Two-sides* in the *New Data Options* box. when we started with the grid. Sometimes, however, an object will appear to have 'holes' which seem to fill as the *Preview* image is moved around in the window. This is because the polygons are not all facing the same way. You can only see single-sided polygons which 'face' the *Camera*. Selective flipping of these polygons may solve the problem, but the most certain method is to make all polygons double-sided. This process was described earlier and an alternative method is described below.

Go to *Polygon/Surface* to pop up the *Change Surface* dialogue panel. All the *Surfaces* are presently labelled *'Default'*. Click the *Double-sided* button, then click *Apply* to close the dialogue panel. The *Preview* window display should now update to show all polygons as double-sided and provide a clear rendition of the extrusion.

We now need to give the *Top* view a little more streamlining like the grid in Layer 5. Do this by selecting the necessary points and dragging them into more appropriate positions as seen in the in the *Top* view. Note that the 3D grid will contain points which superimpose one another in all three views, so lassoing all the points is not advised in this case. You now need to select points on a more individual basis using all three views to locate the one required. Deselect unrequired *Points* with a click of the LMB. After a bit of manipulation, the grid can be made to resemble Layer 5.

We now need to create the inner walls of the wheel wells. At the moment we simply have a cut-away from one side of the body to the other. So, in *Polygon* mode, select the wheel well area so that polygons bordering the cut-away are highlighted. In the *Display* menu, select *Hide Uns* (*Hide Unselected Polygons*). This hides from view all unselected polygons and makes it a lot easier to add details. Now in *Points* mode, select in a clockwise or anticlockwise fashion, the wheel well points that can form 'one section' of a inner wall within one of the wells. Next, using the *Polygon* menu, press the *Make* button. This creates a polygon between the selected points. Note that a polygon is created according to the order the points were selected, so do it in a rotory fashion or the polygon will be very distorted. Make sure the new polygon faces outwards. If not, use *Polygon/Flip* to get it the right way. Repeat this on the other wheel well. The result should be something like Layer 6.





The roughed out car body is now looking pretty good and almost ready for the wonderful *Metaform* treatment. *Metaform* will be found under the *Polygon* menu in the *Subdivide* command. *Metaform* is one of several polygon subdivision routines. However, if you try to *Metaforming* the shell in its present state, you will find that the wheel well edges and the lower edges of the body will be curved inwards. This is because when an object is *Metaformed*, the closer the points, the tighter the curve they produce. At this stage our car body points are pretty well spread out, so the whole side panels will be curved. Do it anyway. It's instructive to run a *Metaform* treatment to observe the effect. Provided you run only one pass, the *Undo* button will get you back to the pre-*Metaform* state.

To improve the final shape of the wheel arches and to bottom edges of the body, there are two strategies to overcome the inward curving. First, more polygons can be created from the large polygons alongside the wells, so that the *Metaformed* edges stay more flush with the body. Single out the wells using the *Hide Uns* tool. A clearer view of the wells can be had using the *Preview Window*. Now, select the polygons which make up the sides of the well, i.e. those which surround the top, front and rear of the wheel. Apply a *Bevel* to these selected polygons using the *Multiply/Bevel* command button. The *Bevel* command has a numeric dialogue box to control the degree of bevelling. You will have to experiment to get the best setting for this job, but keep all values small, say a centimetre or so. We just need a thin strip along the edge.

Each polygon will now have a bevel along each edge. This looks pretty complicated and we really only want the bevel on the outer edges of the polygons. To remove the unwanted bevels, remove their respective polygons by *Welding* their *Points*. This requires some care, because all selected points are combined into one (the last one selected) by this tool. So, in *Points* mode, carefully select the bevelled point, then its original point, in pairs. Now click on the *Weld* button. When you have done this with each unwanted bevel, the wells are ready to *Metaform*. But wait! There are now some two-point polygons (straight lines) where we welded the points. These are extraneous to the model, so you should discard them using the *Polygon Statistics Requester*. In *Polygon* mode, select the *Display* menu. Now click the *Stats* button and the *Polygon Statistics Requester* will pop up. Go to *Two vertex* and click the + button to highlight them. Click on *Cut* to remove them. With a bit of luck, you should now have some edge polygons around the wheel arches like those in Layer 7.

Layer 8 shows the grid after the polygons forming the door/window boundary and the boot line have have been widened and streamlined a little. A flat bottom forming *Polygon* has also been added underneath the body, between the wheel arches. Remember to use the *Display*/*Options*/*Preview* controls to get a proper view of the object. Just use your imagination, it will come out great...believe me it will!

Now before getting down to the *Metaform* tool, we still need to take care of the bottom edges of the body. Here, the strategy is to add some 'rigidity' to the bottom of the car by stencilling in a row of polygons close to the lower edge. See Layer 9 to get the picture. This will prevent the inward curving of the lower edge during the *Metaform*. This stencilling can be done using the *Solid Drill* (*SDrill*) tool with its *Stencil* option. Here's how.

First ,create a *closed* box object in an empty layer. It should be shaped like that shown in Layer 10, so that its upper face will lay parallel to the lower edges of the car body. Use the body as a guide to making the box. Now, position this box in the *background* so that it cuts across the bottom of the *foreground* body shell, a distance of a centimetre or so. To visualise this, place Layer 10 underneath Layer 8. Now use the *Stencil* tool (*Tools/SDrill*) to stamp a strip of polygons into the shell. Depending on the configuration of your particular mesh, this manipulation may create spurious points. Any that you may find should be removed as follows.

Identify and select spurious points using the *Display/Stats* command in *Points* mode. This pops up the *Points Statistics* panel. Pressing the + button against the *0* (point) *Polygons* and *1* (point) *Polygons* will highlight any relevant points in turn. Click *OK*, then just *Cut* them out. You can also detect spurious points using *Polygon*

mode and the *Display/Stats* command. In the *Polygon Statistics* panel, select and then *Cut* out all 1-vertex and 2-vertex *Polygons*.

The car body *Object* should now look something like Layer 9. Now then, click the *Polygon/SubdivMetaform/OK* buttons and see the results. Cool or what!! If you repeat the process with a second click on the *Metaform* tool, you should end up with a rather nice looking body shell. When the body shape is how you want it, use the *Stencil* technique to define the windows and doors using appropriately shaped box objects. Then add some wheels, some swish paintwork and off you go! Easy!

If you have problems with *Metaform*, you may need to clean up the *Object* first. *Metaform* doesn't like sloppy building, so *Merge* points (*Polygon/Merge*) to combine any point occupying the same co-ordinates as another. Also eliminate two point polygons (via *Display/Stats* in *Polygon* mode) and fix any holes by *Making Polygons* from surrounding *Points*.

The second file in the *MetaCarTutorial* directory shows the *Metacar* after the creation of a spoiler and air ducts. The spoiler and air duct were both created by *Bevelling* a few polygons (outwards or inwards respectively) and making a cross connection for the wing before conducting the *Metaform* transformation. The following gives you some idea of what you can create out of a simple box.

A sports car from a box

The Shoe Box MetaCar with Spoiler and Ducts

Tutorial 8: Boolean Operations using Single-sided and Double-sided Polygons

Surface Nomenclature - The Thewlis Interface Rule©

Each type of polygon plays a unique role in *Boolean* operations, depending on the design of the objects involved and the *Boolean* command you invoke. As a consequence, you may get different results from what you intended. The following will help you to understand how *Boolean* operators 'think', so that you can design objects accordingly. As a starter, you must understand the difference between *single-sided* and *double-sided* polygons. If you are uncertain, read the notes on *Polygons* on page 4. You should also understand the difference between an *open* three-dimensional object and a *closed* three-dimensional object.

NewTek's manual leaves a lot to be desired when explaining many LightWave operations. If a topic can be confusing, it sometimes does a good job of making it so! The *Boolean* operator is a typical example. Furthermore, *open* and *closed* 3D objects are passed over for the most part, leaving the reader to figure out what they are him/herself. Perhaps the following notes will help you to visualise the difference.

A good analogy is a ping-pong ball with you as an ant standing on its surface. A ping-pong ball is clearly a 3D object. It is also a *closed* 3D object. You can see only the outer surface, no matter how you move it about in space. The surface is continuous and you can move around on the surface everywhere without seeing a boundary edge (a polygon with a free edge). This property defines a *closed* 3D object.

Now, when a ping-pong ball is hit so hard that it splits, it still looks like a ping-pong ball, but in behaves quite differently. Looking at this object, you'll see a tear in the substance it's made from and as you traverse the surface, you will encounter one or more edges. Now, as an ant at the edge of the tear you can cross over the interface between what was the outside, to the inside of the ball. The ball is no longer closed. It is an *open* 3D object.

A virtual ping-pong ball (*sphere*) could be constructed in *Modeler* and its surface polygons could be made either *single-sided* or *double-sided*. The default *single-sided* polygons facing outwards will allow you to render the ball without any problem. (It would be invisible, of course, if they faced inwards!) However, you might decide to make the polygons *double-sided*, though the rendered view could never reveal it, for the object is *closed*. From the conventional perspective, you can never see inside the object, or view its inner surface. Outwardly, you do not actually know whether the object is solid, like an apple, or hollow, like a real ping-pong ball.

The concepts of 'solid' and 'hollow' objects is an important part of understanding how the *Boolean* operator deals with *single-sided* and *double-sided* polygons. I find that using a (*surface = interface*) principle, it becomes very easy to visualise. In the physical world, interfaces separate phases (i.e. gas, liquid or solid). So, you can readily visualise a LightWave object as a solid structure surrounded by air, where the polygonal *Surface* represents the *interface* between the two phases.

Each change of phase (air to solid, or solid to air) is a *surface,* an *interface*. Either side of the *interface* (*surface*) the medium is *continuously* air or *continuously* solid. Only when we encounter another *interface* (*surface*) does the medium change. Thus, the *surface* of a *single-sided* polygon is one *interface* between air and solid. Air is on the outside and the inside is solid. I've put this idea together as the *Thewlis Interface Rule©*.

Using this rule, it becomes clear why LightWave considers *closed* 3D objects made from *single-sided* polygons as completely solid objects. A *single-sided* polygon has air on one side and solid on the other. The air is on the side to which the polygon 'faces', i.e. the side into which its *Normal* projects. You could think of this as part of the *Thewlis Interface Rule*. You, the observer, are within the air so you are able to see the *Surface* when LightWave renders the polygon. Beneath the surface the object is solid.

Consider next a sphere constructed from *double-sided* polygons. Because we have two *surfaces* back to back, we have two *interfaces*, effectively in the same space. This means that the changes in medium from one face of the polygon to the other are: air to solid to air. Accordingly, this sphere is an infinitesimally thin, solid shell with air on both sides. An exquisite ping-pong ball!

Now, imagine that you are a knife blade slicing through a closed 3D object; again let's think of the sphere as a good example. Every time you cut through a polygon surface, you pass from the *Outside* of the sphere to the *Inside*. If the object was an apple instead of a LightWave sphere, you would pass from outside the apple into its core - from air to solid. Indeed, if you pass through only *one* polygon you will be inside the sphere. Similarly, a *Boolean* knife in the *Subtract* mode slicing completely through the diameter of a single-sided sphere will leave two *solid* hemispheres. The cut surfaces will each be a circular disk , whose polygons adopt the *Surface* name from the surface of the *Boolean* knife.

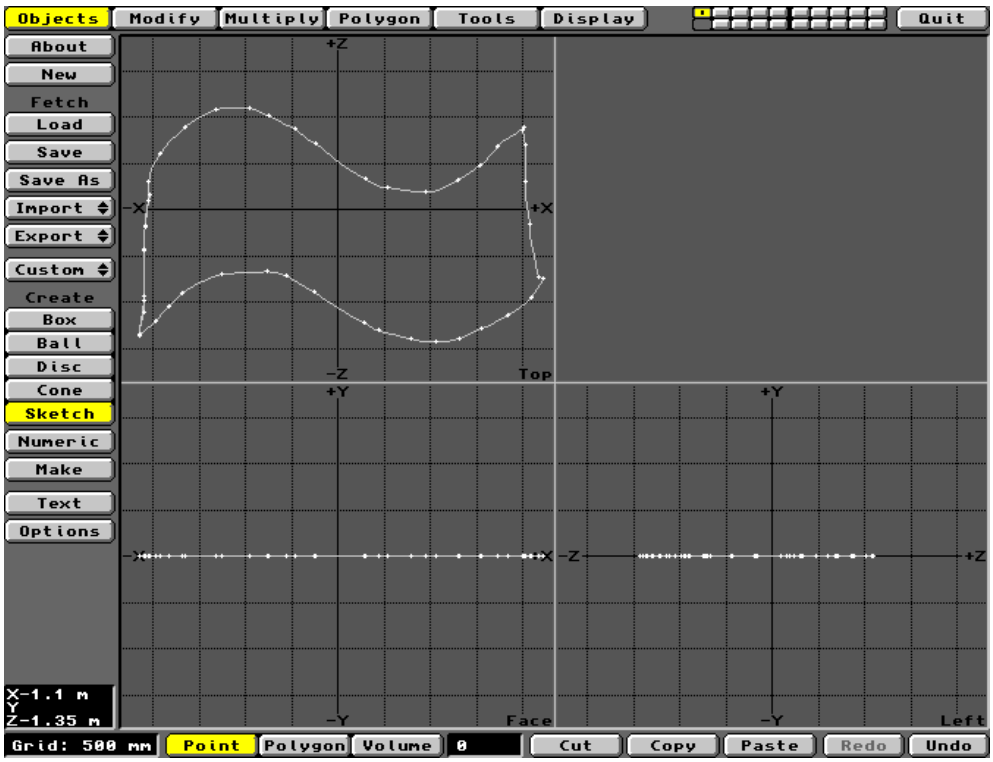
Consider the case of two concentric spheres, one inside the other, with their single-sided polygons facing in opposite directions. The polygons of the smaller sphere face the centre, while those of the larger sphere face away from the centre. You then cut through the outer sphere, and in the process move from air into solid. You continue to cut until you reach the surface polygons of the inner sphere. Here, you will move from solid back into air. Thus, the *Boolean* knife in *Subtract* mode will leave two solid hemispheres, each with a hollow centre, as indicated in the following diagram.

In the diagram below (a), the spheres are represented by circles and the surface *Normals* (arrows) indicate the direction the polygons face. The surface name for the larger sphere is 'L' and that for the smaller sphere is 'S'. As we know from the *Thewlis Interface Rule©*, the arrows always point away from a solid phase into air. The *Boolean* knife object has its surfaces named 'K' and 'E'. The end surfaces 'E' do not actually superimpose any part of the spheres when they are set out in Modeler's layers (b). Thus as the *Boolean* knife cuts (*Subtracts*) the object, new surfaces are created which define the inner walls of the cut. In the resulting object, these walls assume the name of the surface which cut them, i.e. the surface name of the knife where it 'contacted' the object when set out in the front and back layers (c).

The *Thewlis Interface Rule©* applies whether the polygons are *single-sided* or *double-sided*, so to complete this tutorial, consider what would result if the two concentric spheres were composed of double-sided polygons.

Here, the knife passes from air into solid and immediately encounters the second *interface* due to the double-sided polygons of the larger sphere. It therefore passes immediately from solid back into air. As the blade encounters the surface of the inner sphere, it passes from air into solid. Again, the knife immediately encounters the next *interface*, so passing from solid back into air. Completing the cut across the diameter of the spheres, this process is repeated until the knife emerges at the far side of the larger sphere. The *Boolean Subtract* in this case would leave two concentric, hemispherical shells (like empty egg shells) each made from an infinitely thin solid membrane. Both sides of the membrane would be visible when rendered, because each has an outward-facing *Surface*.

When using the *Boolean* functions, be aware that the results will vary depending on the type of polygons being cut. To *Modeler*, it does not actually matter in which direction polygons face. However, to get the results you want, there will be times when you need to model objects with these factors in mind. If the results you get are not what you expected, check the possibility that the polygons are not facing the way they should.



Tutorial 9: The Patch Command

Bedroom Curtains, Billowing Sails and Rather Nice Boats

This Tutorial will help you to understand the use of *Patch*, a powerful tool which creates sheets of *Polygons* across three or four connected *Curves*.

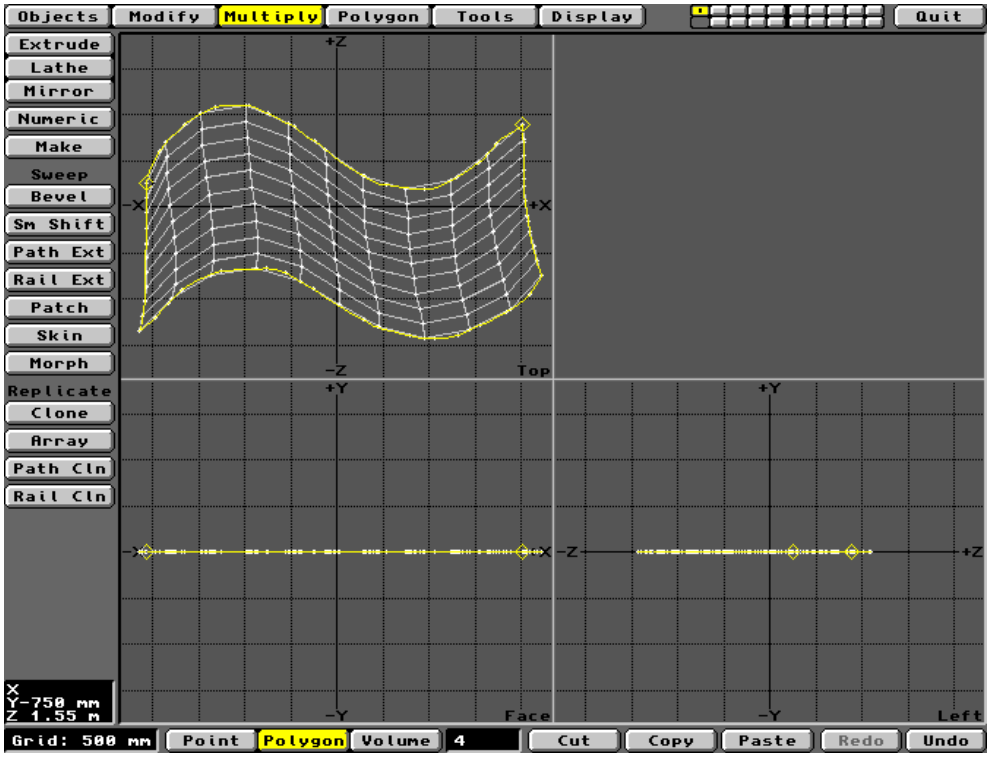
You can create the *Curves* by joining together individual *Points* using the *Tools* menu, or by using the *Sketch* tool in the *Objects* menu. The *Polygon* sheets created will be either two- or three-dimensional in character depending upon how the parent *Curves* are manipulated. Whatever their spacial arrangement, the *Curves* must have overlapping ends or overlapping *Points*, which can be *Merged* or *Welded*, as appropriate.

Curtains, etc.

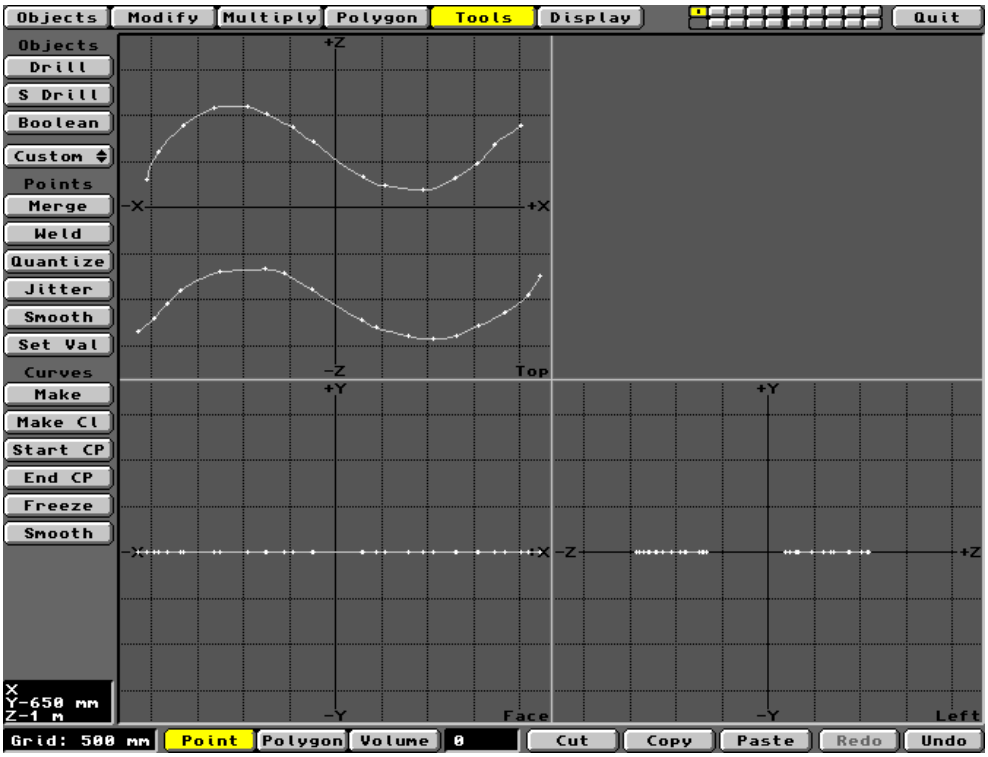
In this first example, four *Curves* will be created in a single *Edit Window* and subjected to the *Patch* operation.

Click on the *Objects* menu and go to the *Sketch* command. Click this, then click *Numeric*. This will pop up the *Draw Freehand* panel. Here, ensure that the *Curve Type* is selected. This will cause any line which is drawn freehand will be converted into a *Spline Curve* rather than a *Polygon*.

Using the pencil cursor, draw a horizontal S-shape in the upper half of the *Top Window*. If you use the LMB, the line will be drawn in a yellow dotted line. Click the *Make* button to complete the drawing. If you sketch the line using the RMB, it will be tranformed into a *Curve* as you release the mouse button.

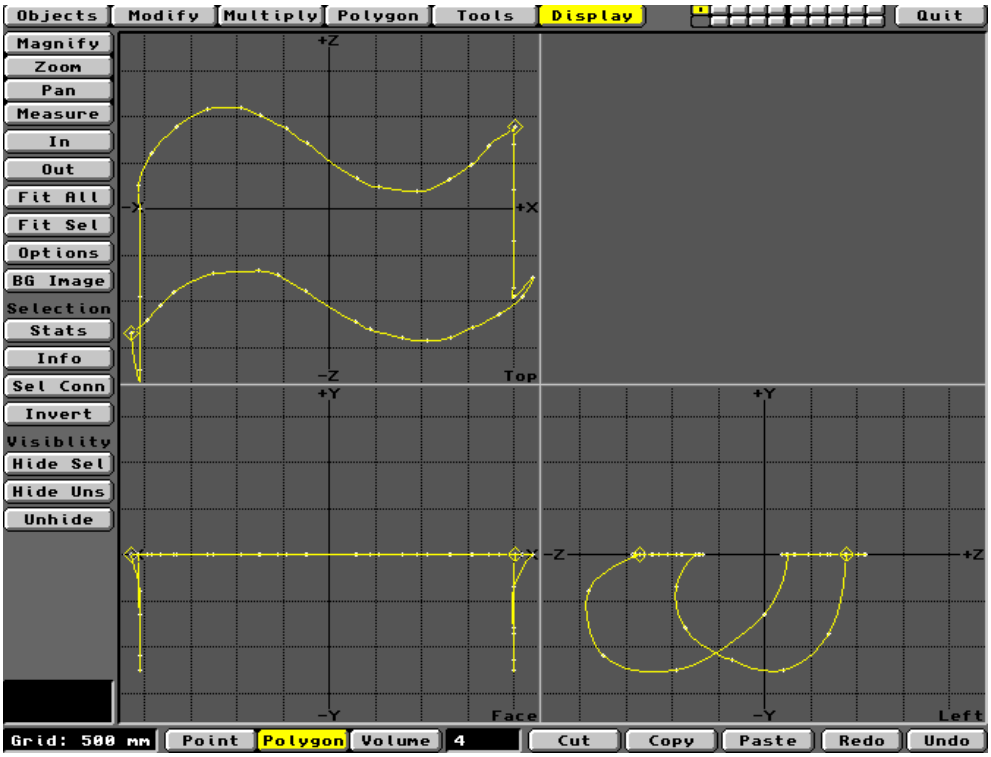


Repeat the drawing in the lower half of the Window. You should have a pair of *Curves* similar to Graphic 1 below.



Graphic 1 The first pair of Curves

Now place the pencil cursor on the left end of the upper *Curve* and draw a line downwards to the left end of the lower *Curve*. Ensure that the cursor is exactly on top of the terminal *Points* in each case. Repeat this with the right hand end of the *Curves*. There should now be four, more-or-less overlapping *Points* at each end of the shape. It will look something like Graphic 2.



Graphic 2 The four overlapping Curves

With the *Points* mode button selected, click on the *Tools* menu and then the *Merge* button. Leave the *Automatic* option selected in the pop up panel and click *OK*. A message will tell you how many *Points* have been *Merged*. There should be four, or perhaps fewer, depending on your drawing accuracy. *OK* the message and attend to any un-*Merged Points*. It should be fairly obvious which pair(s) haven't *Merged*. The sure way to fix these is to select each pair and use the *Weld* command. Only do this with one pair at a time!

Note: When you attempt to *Merge Points*, you may get a persistant message saying that the relevant *Points* must overlap, even when they do. This seems to be a bug and is cured using a patch which was issued with LightWave version 4. Refer to 'Known Bugs' at the end of the Guide. The *Weld* command is unambiguous, but must be done on one overlapping pair at a time, otherwise *all* selected *Points* will be welded into one.

You should now have four *Curves* connected at their ends. Click on the *Polygon* selection mode and all four *Curves* should become selected. If not, use the *Shift*/LMB system to select the rest. You must have all four *Curves* selected for *Patch* to work.

Under the *Multiply* menu, click on *Patch*, to pop up the *Make Spline Patch* panel. This allows you to enter the number of *Knots* or *Points* to be incorporated into the sides of the *Spline Patch*. *Knots* are placed on the *Curves* according to their rate of change in curvature. *Points* are drawn using the *Length* option. This simply divides the length of each pair of *Curves* equally, accordingly the number set. Each pair of *Curves* will then be connected with lines joining corresponding *Knots/Points*. The *Knots/Points* are organised as *Perpendicular* (to the last *Curve* selected) and *Parallel* (to the last *Curve* selected).

Click *OK* and the *Patch* mesh will be generated. It should look something like Graphic 3 (the *Length* option was used in this case).

Graphic 3 The completed Patch

This *Patch* is a two-dimensional mesh which can be modified with any of the *Point* and *Polygon* commands. The *Magnet* tool would be especially useful for generating further curvature in the *Face Window*. Something like this *Patch* could be adapted as *Objects* like curtains, etc. Remember to *Triple* it (*Polygon* menu) if you intend to animate the *Patch* in some way. *Patches* can be incorporated into other groups of *Polygons* for the purpose of *Object* creation.

Sails, etc.

This excercise extends the *Patch* mesh into the third dimension, so that shapes like a billowing sail can be created very easily.

Repeat the first example by drawing two horizontal S-shapes in the *Top Window*. This should look like Graphic1 above.

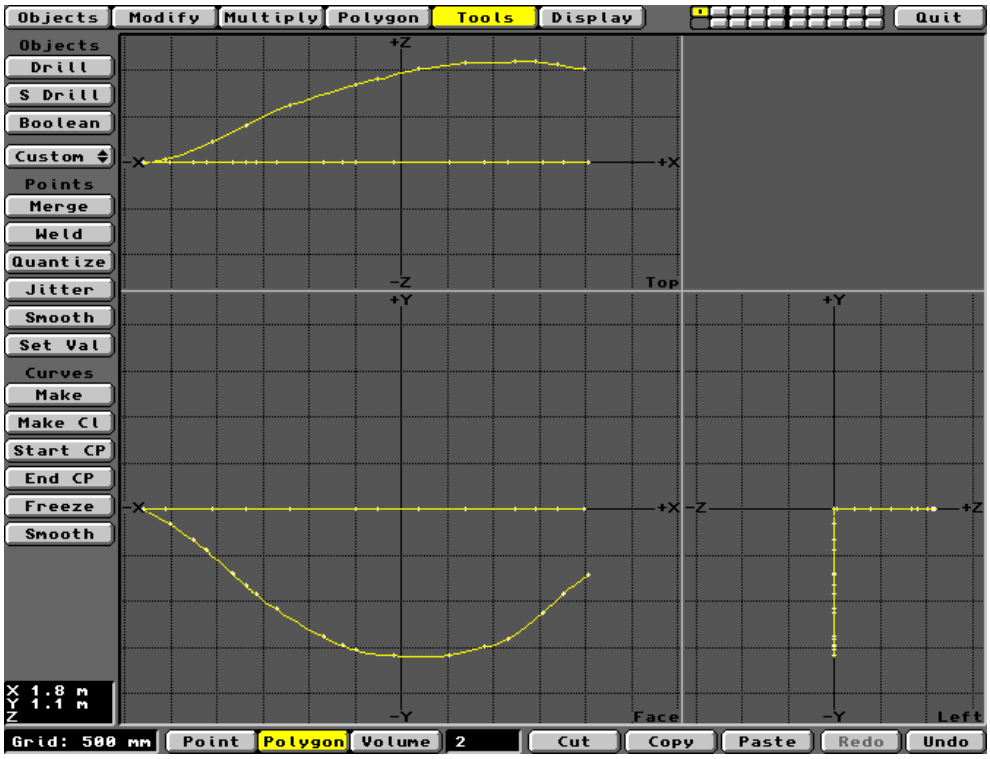
To connect the ends of these *Curves* with two *Curves* drawn in a different dimension, you will need to use the *Points* method. Under the *Polygon* menu, select the *Create/Points* tool. The cursor will change to a gunsight. You can use this to position the location of the cursor in three dimensions.

In the *Top Window*, place the sight over the left hand *Point* at the end of the upper *Curve* and click with the LMB. The cross-hair will remain in position. Now move the cursor into the *Face* (or *Left) Window* and ensure the sight is placed where the graticule crosses the plane of the *Curve*. Click with the LMB to fix the graticule on this location. Now click on the *Create/Make* button. A *Point* will be created.

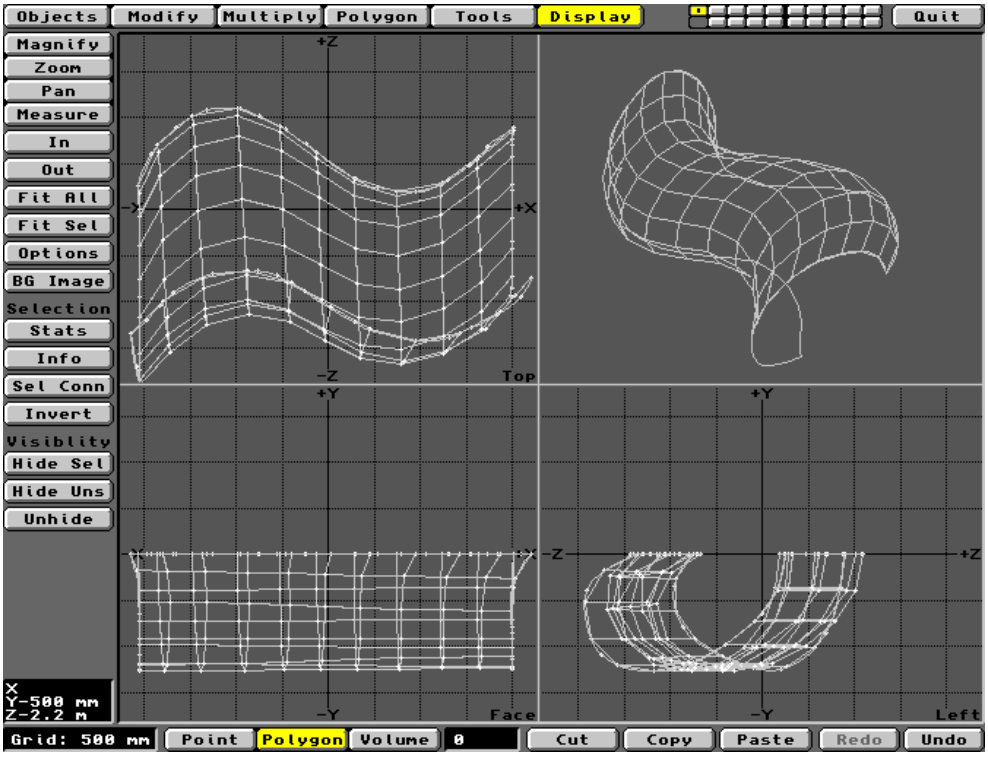
Move the sighting cursor to into the *Left Window* and *Create* a series of *Points* in a curved path, which eventually connects up with the left end of the second *Curve* in the *Top* window. This manipulation may require some practice, but by using the cursor in one or other Window, the *Points* will eventually provide a set of four *Curves* resembling Graphic 4 below.

Graphic 4 Four Curves in two planes

Using the *Merge* and/or *Weld* tools, ensure that each end *Point* of the four *Curves* are connected as shown in Graphic 4. Now go to the *Multiply* menu and click *Patch*. Next, *OK* the required *Knots/Points* settings and the new *Patch* of *Polygons* will be generated. This should look something like Graphic 5, a pretty fair approximation



to a billowing sail. You could of course modify this to taste using the *Point/Polygon* tools.



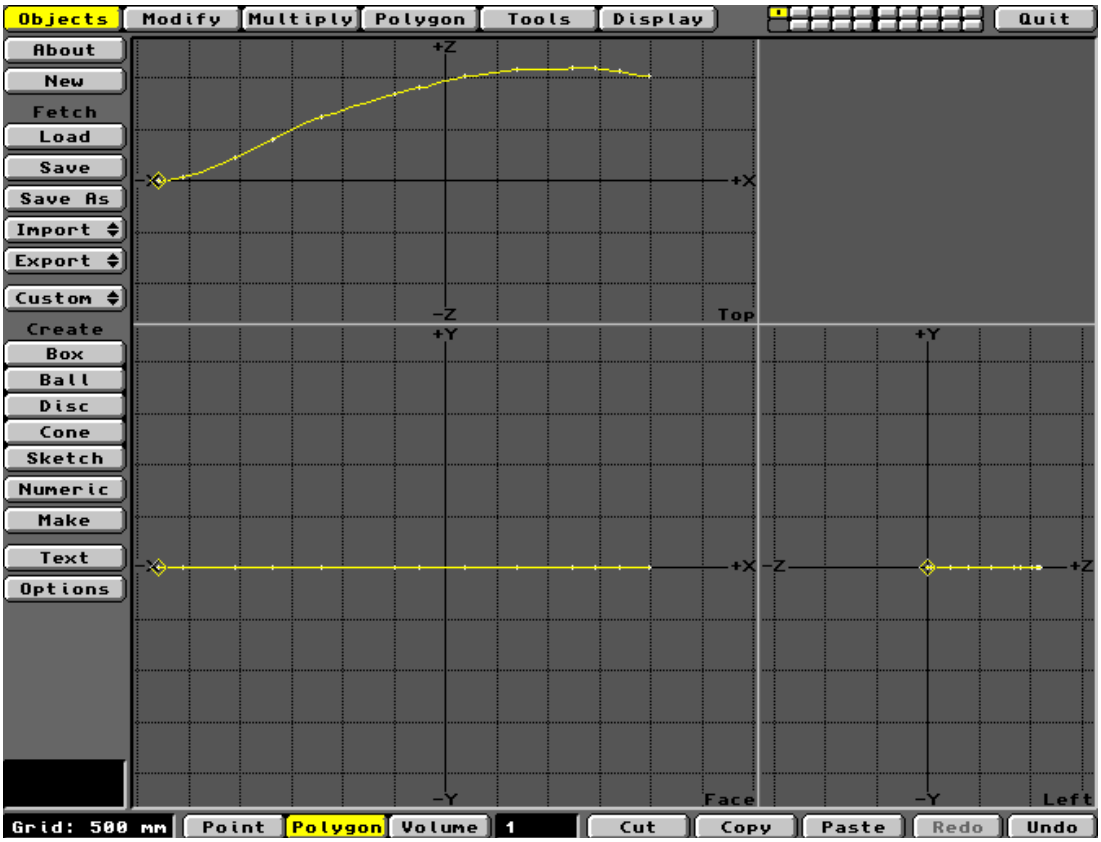
Graphic 5 The billowing sail Patch

The Boat

This excercise uses three *Curves* to generate a three-dimensional *Patch*, easily adapted to a boat or any *Object* which has triple axial curvature.

Consider the shape of a boat's hull. What is the simplest shape that will enable its construction in Modeler? Well, it's got a plane of symmetry running vertically through the keel, so that's the basic shape we need. The hull can then be constructed from two mirror images of this 3D *Patch*. As above, we'll use freehand *Sketching* and *Points* generation to develop the three *Curves* which define our half-hull *Patch*.

In the *Top Window*, *Sketch* a freehand *Curve* to define the shape of the deck as seen from above. Remember this is a half-hull, so you only need one *Curve*. Make sure the starting point is exactly on the X-axis. Later, we'll use the X-axis as a plane of symmetry. The required *Curve* will be similar to Graphic 6 below.



Graphic 6 The deck Curve

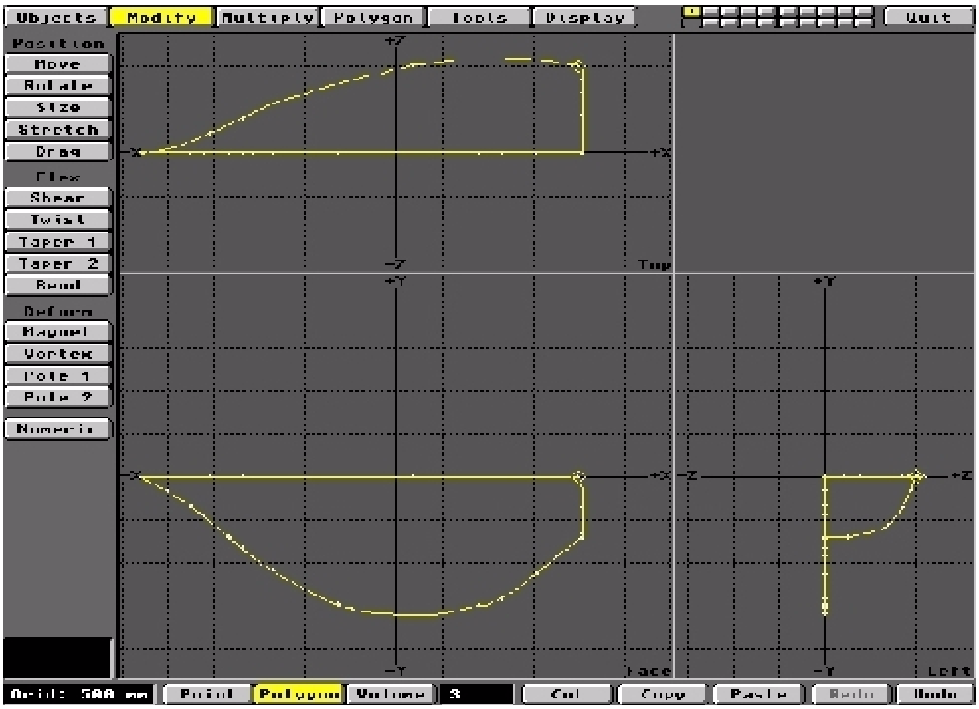
Now *Sketch* the shape of the keel in the *Face Window*, ensuring that the cursor starts exactly at the starting *Point* of the first *Curve*. *Merge* the *Points* at the sharp end and you will get something like Graphic 7.

Graphic 7 The keel Curve is added

The next step is always the most difficult to draw, the third adjoining *Curve*. This *Curve* will be seen face-on in the *Left* view. You must use all three *Windows* to get the cursor in exactly the right location in 3D space. Using the *Polygon/Create/Points* tool, click the gunsight cursor on the last *Point* at the right hand end of the hull *Curve*

in the *Top* view. This gives you the *XZ* location. Now move the cursor into the *Left Window* and place the sight exactly on the graticule created from the first position. Click again to set the third dimension and then click the *Make* button. This *Creates* a *Point* at the chosen spot.

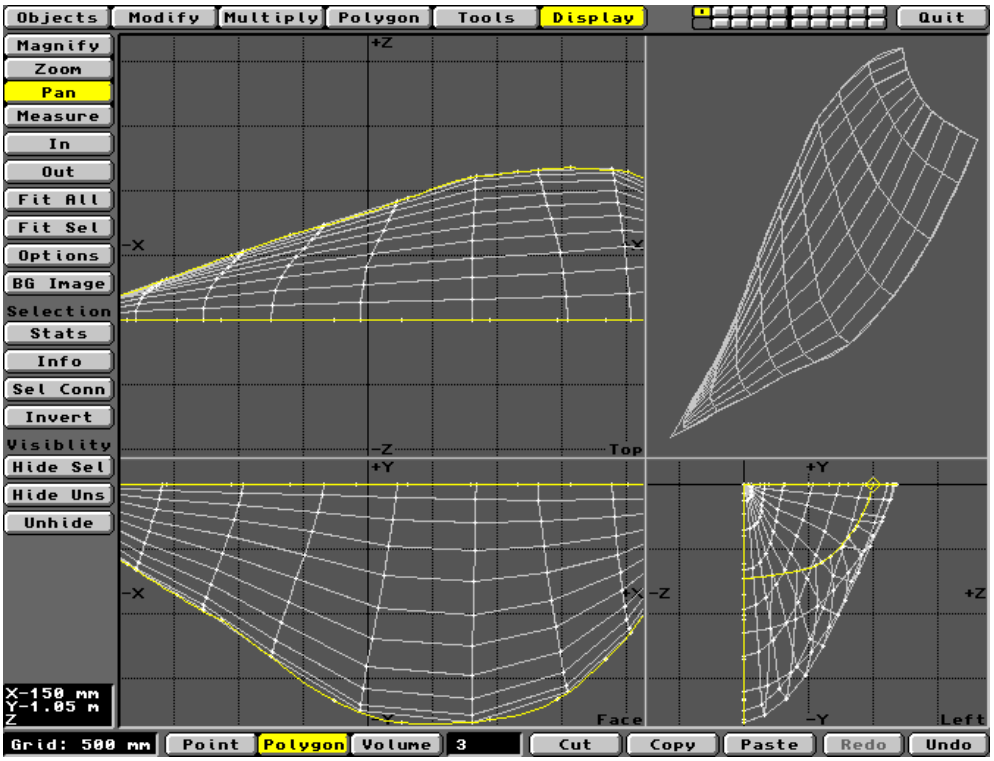
Now move the cursor down a little in the *Left* view to begin setting the *Points* for the hull as seen from the front of the boat. Using the *Polygon/Points/Make* command, set a line of *Points* from the end of the deck *Curve* to the end of the keel *Curve*, building in the cross-section using the *Left* view. After using the *Tools /Curves/Make* command, followed by *Tools/Points/Merge* (or *Weld*), you should have something like Graphic 8 below.



Graphic 8

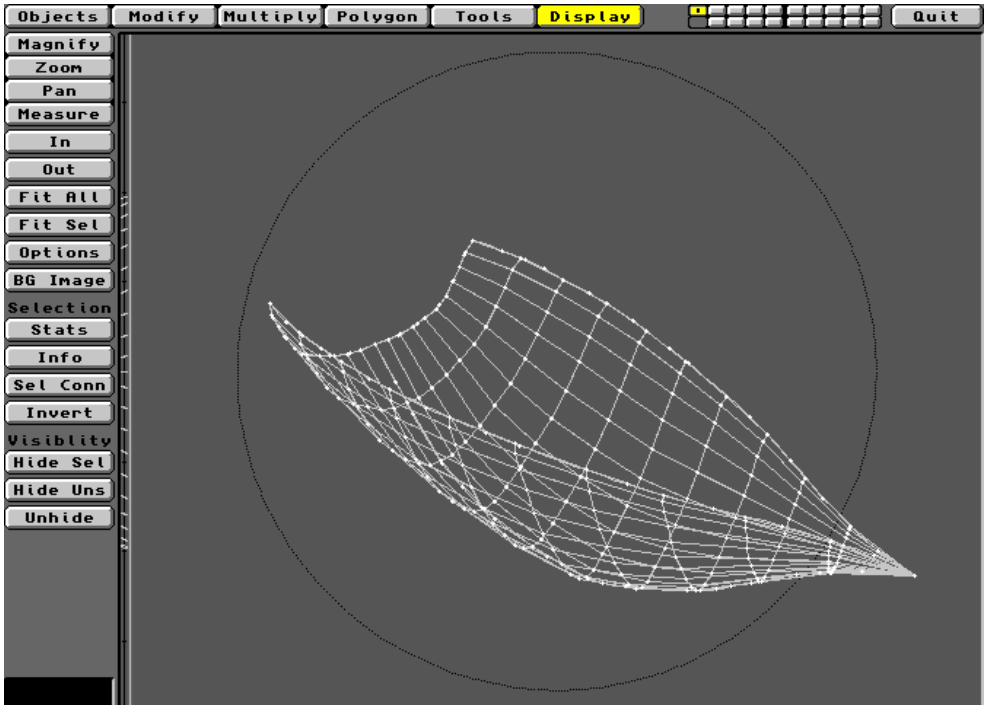
The three Curves merged in three dimensions

Next, invoke *Patch* from the *Multiply* menu and you will generate the half-hull *Patch* as seen in Graphic 9.



Graphic 9 The half hull Patch

To produce the complete hull, we need to generate a mirror image of this *Patch* across the X-plane.
(Note that the above *Patch* was *Move*d away from the X axis to improve the illustration in the *Preview Window*. It should not be moved during the Tutorial).
Click on the *Multiply* menu and then on *Mirror*. Place the *Mirror* cursor on the X-axis and click the LMB. The *Mirror* plane appears along an axis. If it is not running along the X-axis, go to the *Numeric* option. This pops up a dialogue box in which you can set the position of the *Mirror* plane. Select the *X* button. Now, click *Make* to complete the operation. The mesh will obtain its mirror image. To join the two halves together, invoke the *Automatic* mode of the *Points*/*Merge* command. Many *Points* will be eliminated as the halves are fused. The hull will now look like the *Preview* shown in Graphic 10. This *Patch* can now be developed to get the final hull *Object*.



Graphic 10 The basic hull Patch

You can give this hull patch some thickness by making a copy in another layer, reducing its size slightly, then copying it back into the first patch. The smaller patch should sit inside the first with their upper edges level in the side view. The 'thickness' of the hull will then need to be given *Polygons* to enable it to render. This is easily done by hand, because it is a flat surface requiring very few *Polygons*, in fact each edge could provide a single *Polygon* going from stem to stern. However, this large annular *Polygon* would need to be perfectly planar to render successfully. The best plan is to subdivide it into smaller units. The stern end of the hull can be added by constructing a thin box, having a thickness similar to the hull itself. This panel can then be cut into the main body using the *Boolean* operators. Have fun finding which of the routines creates the result you want!

Tutorial 10: The Null Object

Gimbal Gremlins, Shimmering Sheens and a few Head Bangers

Gimbal Lock

One of the most useful applications for the *Null* is in resolving *Gimbal Lock*. This is particularly relevant to any *Object* which has a logical orientation in space, i.e. one with a 'front' and 'back' end and an accepted direction of travel. Typical examples are cars, aeroplanes, spaceships, etc. *Gimbal Lock* (also called *Axis Lock*) is the result of LightWave's method for calculating rotations, though real life gimbals produce exactly the same effect.

The practical result of *Gimbal Lock* is the loss of directional control when the *Object* is rotated 90º (or -90º) in *Pitch*. This is analogous an aeroplane moving from horizontal flight into a vertical dive or a vertical climb. When this happens, the *Heading* (*H*) and *Bank* (*B*) axes of the *Object* become the same. Both controls cause changes in the (*B*) orientation of the *Object*. In essence, the *Heading* (*H*) control is lost. This is easily demonstrated by loading an aeroplane model into Layout. A nice Boing 747 mesh will be found on the Tutorials Disk.

When loaded, you'll see that the plane is facing into the *+Z* axis. This is the recommended orientation whenever you create such *Objects* in Modeler. Ensure that the 'front' of the *Object* faces towards *+Z* and that the *Origin* is located near the logical 'middle' of the *Object*, or within its base if there is one. Select the *Camera View* and using the *Edit/Camera* controls, *Move/Rotate* it until you have a nice oblique view of the 747.

To appreciate the problem of *Gimbal Lock*, you should imagine that you are inside the cockpit (flight-deck if you must) and you are flying the plane. Click on *Edit/Object*, and *Rotate* the *Heading* (*H*) of the 747 using the LMB. It's best to deselect the *Pitch* (*P*) and *Bank* (*B*) buttons, so only the *Heading* is affected. You'll readily see that *Heading* controls the aircraft's rotation from left to right as seen from the

cockpit, I mean the flight deck. The 747 *Object* is actually turning horizontally about its *Origin* (*Pivot Point*).

Go back to the *Rotate* button and set the *Pitch* to 90º. The best way to do this is to click on *Numeric Input* and type the value into the pop up *Object Direction* panel. When you *OK* this, the 747 will take a vertical nose-dive. Now go back to the *Rotate/H* control and try to steer the plane. You can't! All you can do is roll it, exactly what you get from the *Bank* (*B*) button (remember *Bank* is controlled via the RMB). This is *Gimbal Lock.* The closer the *Pitch* gets to 90º, the more the gimbal locks. If you must place an *Object* like a plane in this orientation, you cannot avoid steering loss! Well, actually you can, by using the *Null Object*.

Reset the 747 to its as-loaded position. Do this using the *Object Direction* panel, as above, or simply *Replace* the *Object* with itself via the *Objects Editor*. Click the *Objects* menu button to get the *Objects Editor*. Here, at top right, you'll see the *Add Null Object* button. Click on this once and the *Null* will load. You can see from the *Current Object* information window that the *NullObject* has one *Point* and no *Polygons*. Click *Continue* to see the *NullObject*. It's the six-pointed axis located on the Layout *Origin* and coincident with that of the 747.

The trick to keeping control in the vertical dive is to use the *Null* as the *Parent* for the 747. Select the 747 and click on the *Parent* button at the bottom of the interface. Using the scroll bar, place *NullObject* in the *Parent Object* panel which pops up. *OK* this to get back to the 747. Now you must *Create Key* (*All Items*) at *Frame* zero before applying any movement or rotation from the default position.

To put the 747 into the nose-dive, you must now select the *NullObject* and enter 90º into its *Pitch* setting. This forces the 747 into the same orientation as the *Null*, even though you have not used the aircraft's own *Pitch* control. Now select the 747 and test the steering. The *Heading* control works normally! The gimbal is no longer locked, so the 747's *H*, *P* and *B* controls operate normally. From here on, however, you must use the *Null* to control the 747's movement, but the 747 to control its own rotation. If you offset a parented *Object* some distance from the *Null*, you can rotate the *Null* to make the *Object* follow an orbital path. Often, combinations of *Object* and parent *Null* rotations can be employed to achieve interesting effects.

Rotating Reflection Maps

Animated three-dimensional text is often given a reflection *Image* to add that touch of class. Moving sheens are usually achieved by traversing one or other axis of the reflective *Text Object* with an *Image Map*. However, there are occasions when the sheen must rotate across the surface of *Text*. This is another case where the *NullObject* comes into its own. Here's how to do it.

After your *Text Object* has been generated along with any appropriate *Bevels*, you should find a suitable *Reflection Image.* This should be quite contrasty, such as white clouds in a dark blue sky. Alternatively, you

could take the *Fractal Reflections* image supplied with LightWave and increase the contrast using a paint package. Whatever you choose, load the image file into Layout using the *Images* menu button. Ensure the latent animation has enough *Frames* using the *Scene* editor. Go to the *Surfaces* menu and click on the *Reflection Image* scroll bar and insert the selected *Image*. The *Text Object* should be given a suitable *Surface* colour and then attributes something like the following.

Diffuse Level	20 - 30%
Specular Level	40 - 60%
Glossiness	Low
Reflectivity	70 - 90%
Smoothing	On
Max Smoothing Angle	20 - 30°

Keep all other *Surface* values at the defaults. Now with the *Text Object* in position, add a *NullObject*. Next, *Parent* the *Text* to the *Null*. Then, *Parent* the *Camera* to the *Null*. Move the *Camera* to any desired location so that the *Text Object* is in full view. Next, *Create Key* at *Frame* zero for all items.

We now come to the interesting bit. Ensure you have the *Camera View* selected. To rotate the sheen around the *Text*, you will actually rotate the *Text*, so 'moving' the *Reflection Image* relative to the Text. However, you must do this by *Rotating* the *Null*. This will also cause the *Camera* to rotate in synchrony with the *Text,* since both are *Parented* to the *Null*. Select the *Text Object* and apply a rotation, for example in *Bank* (*B*). This can be any angle you like, but make it large enough for the sheen to be impressive. When you have made sufficient rotation, *Create Key* at a suitable *Frame* number, say 100-150, depending on the duration of the animation. When you test this using the slider button, the apparent result is the rotation of the Layout *Grid*, while the *Text* remains stationary. In reality, the *Grid* is stationary and everything else is rotating together. It's like being on board the Space Shuttle and capturing the Hubble Telescope, while the Earth below drifts slowly by. It's all down to relativity. You can check this by looking at the *XY*, *XZ* and *ZY Views* and scrolling the *Frames*. Clever or what! If you want the animation to loop, ensure that the *NullObject* rotation begins and ends at the same settings.

Use the *Spline* controls to refine the motion of the sheen across the *Text Object*. A positive setting for the *Null's Heading* and *Bank* will cause the sheen to traverse from right to left, while a negative setting has the opposite effect. The sheen is probably most effective with a black *Backdrop*, but you are not restricted in this. Any image or brush can be loaded into the *Images Editor* and then used as a *Background Image* via the *Effects Editor*.

Offset Control using the Null

Any movement or rotation of the *Null* is reflected by all *Objects Parented* to it. This is a very powerful function that can be used to generate legions of *Objects* acting synchronously. More mature LightWavers may remember the video which accompanied the release of 'The Wall' album by Pink Floyd. No? Well never mind. Anyway, it contained lots of animation and one sequence involved hundreds of hammers marching together and striking their heads on the ground (head-bangers I suppose). Well, you can create exactly the same effect using the *NullObject*. In the *3D:Objects/Tools* directory, you'll find the *Hammer Object*. Load this into Layout. As loaded, the *Hammer* 'faces' away from the *Camera*, into the *+Z* axis. Select the *Hammer* and use *Rotate/H* to get it facing to the left. Note that its axis (*Pivot Point*) is located in the end of the handle on the *XZ* plane.

Go to the Objects Editor and click the Add Null Object button. Back in Layout, you'll see that the *Null* is coincident with the *Hammer's* axis, at the bottom of the handle. This is a nice point for rotating the *Hammer*, so let things stay as is. Select *Hammer* and make the *Null* the *Parent*. *Create Key* at *Frame* zero for *All Items*. Now go to the *XZ View* so you can see the set-up from above. Now click the *Objects* menu button to pop up the *Objects Editor* and here, *Clone* the *Hammer* five-fold. Using the *Current Object* selector, highlight each clone *Hammer* in turn and *Move* it sideways along the *Z* axis (deselect *X* and *Y* if necessary). Repeat this with each *Clone*, positioning them in a neat row. You should now have a row of six *Hammers* along the *Z* axis with the *Null* at the Layout *Origin*.

Go back to the *Camera View* and *Rotate* the *Null* in *Bank* (*B*) to about 85°. All six *Hammers* should rotate anticlockwise in perfect synchrony. In the default 30 frame *Scene* setting, *Create Key* at *Frame* 15, with the *Hammers* in this position. *Create* a *Key* at *Frame* 30 identical to *Frame* zero. This will ensure a looping action. Run a trial *Preview* to ensure things look OK.

Next, you can either *Save* this *Scene* as usual in the *3D:Scenes* file, or simply save it to RAM: for the next step. To save a temporary *Scene* in Ram:, just type Ram: into the *Path* field of the *Save Scene File* requester. Call it 'Hammers'. Keep the *Scene* open. Go to the *Objects Editor* and click the *Load From Scene* button. This will *Load* all the *Hammer Objects* and the *Null* again using the *Hammers* file (from *Ram*: or *3D:Scenes*, according to your previous action). Don't load the *Lights* from *Scene*. All six *Hammers* plus the *Null* will be duplicated. They are located exactly where the originals were located, so each hammer is now two hammers. The clone *Null* is labelled *NullObject(2)*.

Using the *XZ View*, select *NullObject(2)* and *Move* it a little distance along the *X* axis. This should provide you with a second, identical row of *Hammers*. *Create Key* at *Frame* zero for *All Items*. Notice that NullObject(2) is showing a *Motion Path*. This is because it had an identical setting to the original *Null* placed at the Layout *Origin*. Select its *Next Key Frame* (15) and *Move* the *Null(2)* to its starting location along the *X* axis. Again, *Create Key* at *Frame* 15. Go back to *Frame* zero and *Create Key* for *Frame* 30, so that the two are identical. This ensures the looping action.
Go to the *Camera View*, repositioning and re-*Keying* it if necessary, then scroll through the *Frames*. The new row of *Hammers* should mimic the movement of the first over the thirty *Frames*. Repeat the *Load From Scene* step as many times as you want, moving each row of *Hammers* along the *X* axis (using their respective *Null*) until you have an army of them. Test the set-up with a *Preview* render. You can refine the motion of the *Hammers* using *Splines* and the *Envelope Editor*. However, any such tweaking is best done before the *Cloning* step to save work, since any *Splines*, etc. are also cloned. The resulting animation will be quite like the Pink Floyd video. It really will, believe me!

These examples illustrate some of the powerful properties of the *NullObject*. You can use it to bring about 'global' effects on some or all *Objects* in a *Scene*. Since the *Null* can be scaled using the *Size* control, any *Objects Parented* to it can be scaled simultaneously. Alternatively, you could rescale some or all *Objects* in one or two dimensions only using the *Stretch* command on their *Parent Null*. The more you consider its potential, the more you will recognise the value of the *Null* in your animations. Unfortunately, there is almost no mention of the *NullObject* in the official LightWave manuals. Not surprisingly, it's the most under-used feature in LightWave 3D.

Tutorial 11: Bevelled Text Objects

I Did It My Way!

The construction of *Bevelled Text* is important when designing company logos, video titling, etc. However, this task can present problems if not carried out with care. Typically, the construction of these objects involves a combination of the *Bevel* and *Extrude* commands on 2D *Text Polygons*. Details of the *Bevel* operation are given in the discussion of the *Multiply* menu in Modeler. Here, it will be noted that polygons subjected to the *Bevel* operation are moved from their initial location by an amount determined by the *Shift* value. They are also shrunk according to the *Inset* parameter. Both operators will accept positive or negative values. The application of *Extrude* and *Bevel* may result in a *Text Object* that renders with faults in the outer surfaces. These usually appear as grooves crossing from the front surface of the text to the back. This result is due to the effect on the text's face polygons by an inappropriate *Inset* value after the extrusion operation. The NewTek manuals offer varied advice on rendering *Bevelled Text Objects*, but I have found them confusing to say the least. The results seem to be affected by the order in which the two commands are used. After looking at all the options, I believe the following approach is most satisfactory.

In Modeler, click on the *Text* button, found under the *Objects* menu. The *Generate Text* panel pops up and this should be loaded with your desired font. The *Font* scroll bar will probably indicate *(none)*, so click the adjacent *Load* button to activate font loading. The contents of LightWave's default *Fonts* directory will appear in the pop up requester. Select the desired *Font* from the list. *OK* the selection and the *Font* scroll bar will confirm that the font has loaded. A number of *Fonts* can be *Loaded* in this way, depending on free memory. You can remove any or all *Fonts* from memory by clicking the *Remove* button for each *Font* in turn. For this tutorial, we'll use a nice chunky one such as *SansCondLH-HeavyBold*, but almost any font will do.

The *Corners* buttons provide the options *Sharp* or *Buffered*. These determine how many *Points* will be used to construct the corners of the *Text Object*. Usually, the extra *Points* provided by the *Buffered* option are not required and the default *Sharp* option is satisfactory. You can explore each option later, but it's worth noting here that points near the corners of *Text Objects* are often responsible for poor results. See below for further details.

In the *Text* field, type in your desired letter, word or phrase. For our purpose, let's simply type capital 'J' since it's got both straight and curved elements. Click the *OK* button and the letter 'J ' will be constructed in the *Face* window. Adjust the visual size of the letter by using zoom *In* or *Fit All* under the *Display* menu. You will see that the letter is a two dimensional shape. The *Top* and *Left* windows show it as a single line bearing the construction *Points*. In the *Face* view, notice that the inner curve of the letter contains a lot of closely spaced *Points*. If RAM is limited, you might feel it useful to reduce these to a smaller number without radically affecting the curvature. Do this by getting really close to the curve using the zoom *In* and *Pan* buttons. Now, using *Point* selection, click on each extraneous *Point* to select it and click *Cut* to remove it. If you've got RAM to spare, you can forget this economy measure altogether. Just remember that lots of curved letters consume more RAM then 'straight' letters.

If you used the *Sharp* option for *Corners*, the corners of the letter will have a single *Point*. If you used the *Buffered* option, extra *Points* will be inserted on each side of the corners. These extra *Points* might be useful if you plan something special like text morphing, but they are usually not required and simply consume RAM. It is therefore usual to use *Sharp Corners* when rendering bevelled text. After any necessary tidying up, click the *Fit All* button (*Display* menu) to get everything in view.

Click on *Polygon* selection, so that you can highlight the text face polygon by clicking on one of its edges with the cursor. This done, you'll see that the letter is a single *Polygon*, with its surface *Normal* directed towards you in the *Face* view. If the *Normal* is directed away from you, use the *Flip* command (*Polygon* menu) to ensure it points in the *-Z* direction. Now, keep the letter highlighted and let's get on with the bevelling.

Open the *Multiply* menu and click the *Bevel* button. This pops up the *Bevel* data panel, into which you must enter values for *Inset* and *Shift*. These two values determine the physical size of the bevel which will be generated along the outer edges of the text polygon. Logically, you cannot create a bevel which exceeds the size of the letter itself. Well, in LightWave you can though the result is probably unusable, but do experiment when you can properly interpret the results. Now, before entering any values in the data fields, examine the notional size of the text polygon. Compare it with the *Grid*, so you can determine the most appropriate *Units* to use with the values. These control the size of the bevel. As a starting point, consider bevel values around one tenth the physical width of the letters. So, if the *Grid* indicates the letter is a few centimetres wide, adjust the *Units* field to millimetres (*mm*) using the (+) and (-) buttons.

The letter 'J' will be seen to be about 15 centimetres wide, so enter an *Inset* value of 10mm and a *Shift* value of 10mm (or if you prefer centimetres, 1cm each). Click *OK* to construct the *Bevel*. The highlighted polygon will

have moved away from its initial position and will be shrunk in size. You will see from the *Top* and *Left* views that it has been *Shifted* 10mm (1cm) in the direction of the *Normal*. From the *Face* view, it has been shrunk inwards by 10mm all round, the *Inset* value. A series of new polygons has been created along the edges of the face polygon. These new polygons form the bevel. The outer dimensions and location of the bevel equate to those of the original face polygon. At this stage, it's worth looking closely at the bevel borders, particularly around the corners of the letter. When the area around a corner is bevelled, the *Points* are often moved 'across' the corner *Point*, causing the bevelled border sections 'overlap' at the corner. You can easily see this by examining the border in *Front* face close-up around the corners of the letters. Most important is the corner *Point* which you will easily recognise in close-up. If the borders appear to overlap the corner, there will be several *Points* that are clearly in the wrong position, having been shifted away from their original position on either side of the corner. Select the offending *Points* and using the *Drag* tool, move each one back into line within the border sections. When this is done, the bevels will render perfectly. The letter will now render as bevelled text providing it is viewed exactly face on, but you will usually require the letters to have 'thickness'. This is where the *Extrude* command is employed.

Before extruding the letter, it is helpful to assign *Surface* names. The extrusion process usually creates complex lattices and it can be difficult to identify faces, bevels, sides, etc. Ensure the face polygon is highlighted. If it's not highlighted, do so by clicking the cursor on one of its edges. Note that the attached bevel polygon will highlight as well, because they share the edge. Deselect the bevel via a separate click on its outer edge. Open the *Polygon* menu and click the *Surface* button. The name '*Default*' can now be replaced with a suitable name. '*Face*' would seem appropriate. Click *Apply* or hit the *Return* key to apply the new name. If you intend to treat the bevel as a separate surface, name it now, but leave the *'Face'* polygon highlighted.

The easiest way to name the bevel is to click the *Hide Sel* (*Hide Selected*) button under the *Display* menu. This will hide the highlighted *Face* and leave you with only the bevel polygons. The appearance of the object will be unchanged, but the inner face is now hidden. You can prove this by clicking on the inner edge of a bevel polygon. Only the bevel will highlight, proving that the face polygon is inaccessible. Deselect the highlighted edge polygon so the whole object is now active. Go back to the *Polygon* menu and use the *Surface* button to name the object '*Bevel*'. Go back to the *Display* menu and click the *Unhide* button to recover the face polygon.

We will now extrude the bevelled face to give the letter 'J' some thickness. Open the *Multiply* menu and click *Extrude*. This changes the cursor to the extrude arrow, which can be used to create the extrusion manually. If this is your method of choice, single-click the cursor onto the *Face* view. The text object will be fitted inside a dotted-line bounding box and the *Top* and *Left* windows will show an extrusion gauge (T-bar). Drag this bar with the cursor until the desired thickness is achieved, then release the mouse button. To complete the process, click the *Make* button. The extrusion process generates identical face and bevel polygons to the rear of the object. You can use the *Numeric* button to perform the extrusion more accurately. Type in a suitable value, say 15mm (1.5cm). A single *Segment* is sufficient (as is obtained with the manual extrude method), but you can split the extrusion into as many slices as you like, RAM permitting. More importantly, ensure you extrude along the *Z* axis! Deactivate the *Extrude* tool by clicking another menu button.

Use the cursor to select the letter's face polygon in the *Face* window. Remember to deselect the connected bevel polygon. In the *Top* and *Left* views, you will see that the two *Face* polygons both point outwards. LightWave has flipped the rear *Face* polygon automatically, because this is the logical requirement. The same thing has happened to the rear bevels, though they presently slope inwards rather than outwards. This is the logical result of the extrusion process, it simply reflects the shape of the original object. If you do not intend to render the rear face of the text, you can leave things as they are. However, if you wish the rear bevel to look like the front bevel, the rear face polygons will have to be moved out a little. Let's do this now.

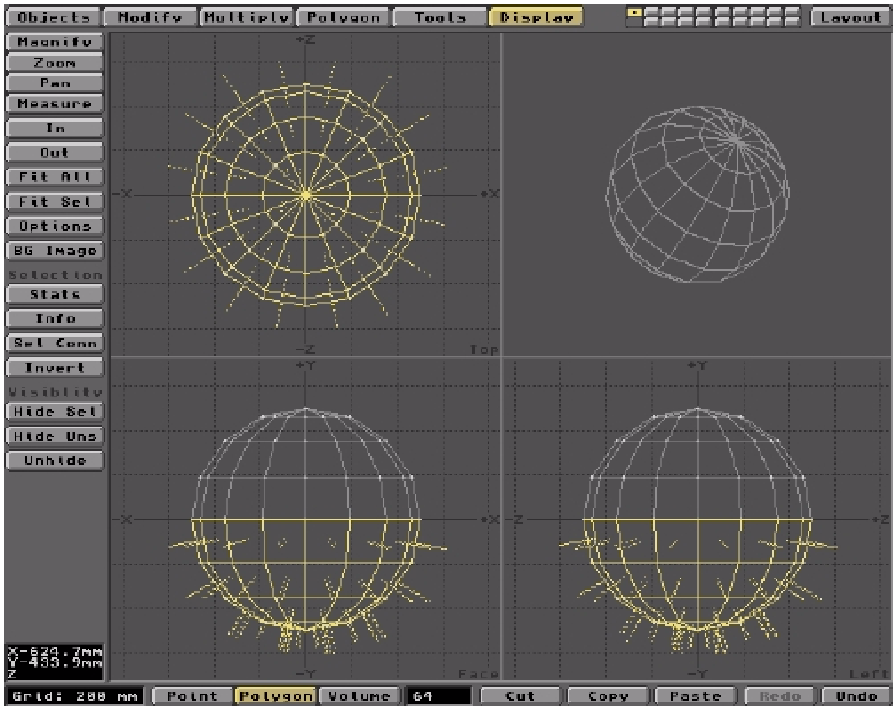
Select the text face polygon by clicking on an inner edge in the *Face* view. Deselect the attached bevel polygon. In the *Top* view, you will see that both face polygons are highlighted. Actually, we need to displace the rear face towards *+Z* by twice the *Shift* value. So, deselect the front face polygon by clicking on it in the *Top* view. We now have only the rear face polygon selected. Open the *Modify* menu and click the *Move* button. Don't try to move the face manually, because it's not that easy. Open the *Numeric* panel instead. Here, enter a value of 20mm (2cm) in the *Offset Z* field. *OK* this and the rear face will be placed in exactly the correct position. Deselect the rear face and open the *Display* menu.

Click the *Stats* button to pop up the *Polygon Statistics* panel. Here, you can check the location of the named *Surfaces*. Click on the scroll bar to find surfaces with a particular name. Select a name, then click on the '*with Surfaces*'(+)' button. All polygons assigned that name will be highlighted. Try it with '*Face'*. You should see the two highlighted face polygons which outline the letter 'J'. Now scroll to '*Bevel*'. All the '*Bevel*' surfaces will be highlighted. Note that these include the extruded text's 'side' polygons, because these were produced from the *Bevel* itself. If you want to give these side polygons a different surface appearance, they must be given a different name. To do this, deselect everything and put all your attention to the *Top* and *Left* views of the text object.

Maximise your view of the side polygons by dragging out the windows and clicking *Fit All* in the *Display* menu. Our interest lies in the polygons with parallel edges joining the front and rear bevels. Select all these by dragging the mouse over them, holding down the LMB and the *Shift* key. Avoid touching the bevels. Ensure all the side polygons are highlighted by traversing the field slowly, back and forth. When everything looks OK in the *Top* and *Left* windows, go to the *Surface* button under the *Polygon* menu and rename these polygons '*Side*'. The bevelled text object is now complete, so save it to an appropriate location with a suitable filename, how about '*Bevelled J*' ! To render this object with an impressive chrome type reflection, you will need a nice reflection image. Unfortunately, none is included with v3.5 so you'll find one I made for myself on the WaveGuideTutorial Disk, together with an example render. This was produced by applying the *RipplingChrome* surface to the face polygons and using appropriate parameters in the *Bump Map* texture panel. Getting this exactly right is very much trial and error, so keep trying with slightly different settings. If your renders don't turn out so well, despite a lot of effort, see the note below.

Important Note

Textured surfaces render best on triangular polygons. If your render shows blocks of apparently untextured surface, you should *Triple* the relevant *Polygons*. This problem is particularly obvious with very reflective finishes such as the *Chrome* attribute. Reload the text into Modeler , select the offending surface and use the *Triple* command under the *Polygon* menu. This will fix all such cases of poor *Surface* rendering.



Tutorial 12: How to create a perfect Boing Ball

Boings are easy, once you know how!

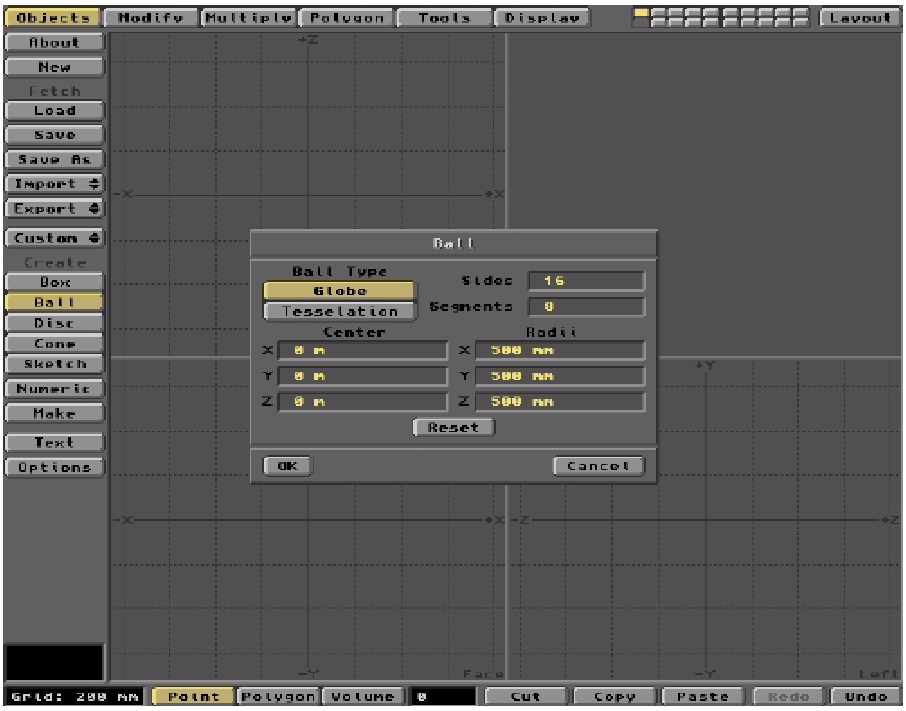
Here's a step by step guide to creating a 16x8 (16 side, 8 segment) presentation of the defacto Amiga logo. You can adapt the instructions to create any other presentation you fancy.

You're talking Balls!

The first thing to do is generate a very crude Boing Ball with the desired number of sides and segments. By this, we mean the number of coloured sections making up the 'longitudinal' aspect (sides) and the number of 'slices' or segments from 'north' to 'south'.

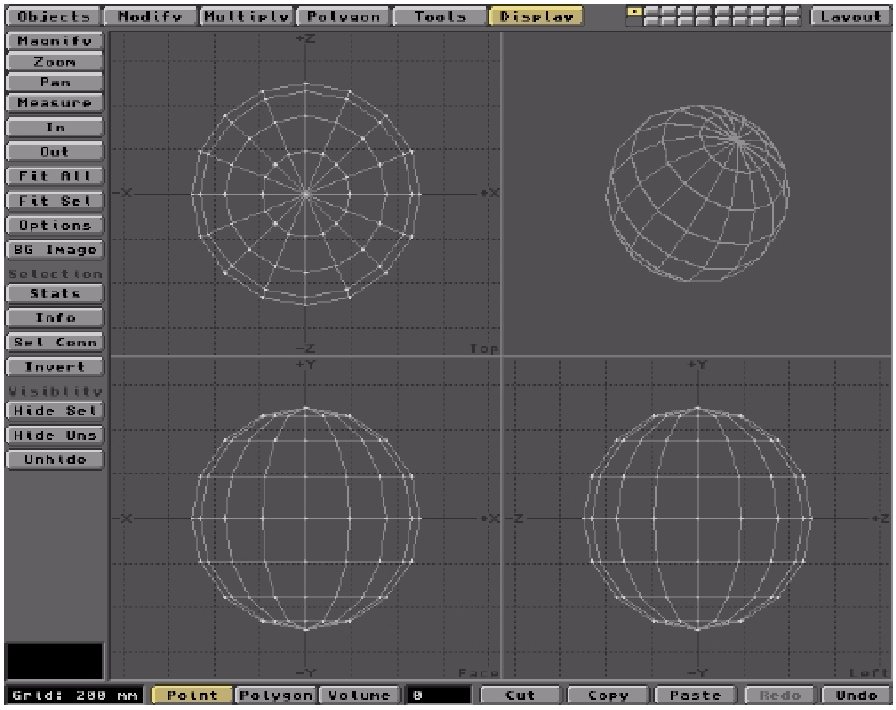
You do this by going into Modeler and clicking on the Objects menu, Create/Ball tool. This will provide you with a 'ball-creation' cursor. You can drag out a bounding box using this cursor in the View Windows and thereby create the basis for a ball. But don't! Instead, click on the Numeric button.

This pops up the Ball parameters panel, shown below.



Here, you can set the ball's structural Type, its number of Sides and Segments, its X,Y,Z dimensions (Radii). The default 'Globe' type is the one you need. It's divided 'horizontally' and 'vertically'. Experiment with the tesselated version to see the difference. If you experimented with the ball-creation cursor, the panel will display the values you generated with it. Just press the Reset button to get things back to the defaults.

Now, enter the desired number of Sides (16) and Segments (8) in the appropriate fields. If you prefer the classic 12x6 Boing format or any other combination, just enter the relevant numbers. Leave the default Center at 0,0,0. That's the centre of LightWave's grid system for both Modeler and Layout, the logical place to put the Object you'll create. You can make the X,Y,Z dimensions (Radii) any size you like, providing they're all the same. It is a perfect ball we want after all! The defaults will be probably be fine. OK the settings to close the panel and get back to the Modeler windows. You'll see the bounding box/cursor has set everything up and awaits your approval. So, click the Make button to complete the construction. The box/cursor will remain on screen along with the generated Ball. Free up the cursor by clicking on one of the the Menu button at the top of the screen. You should now see something like this....



In this graphic, I've activated the 3D view window (top right) with a Static Solid presentation. The object has been rotated on screen with the mouse to give it a classical Boing 'pose'. It often helps to see what you've created in 3D.

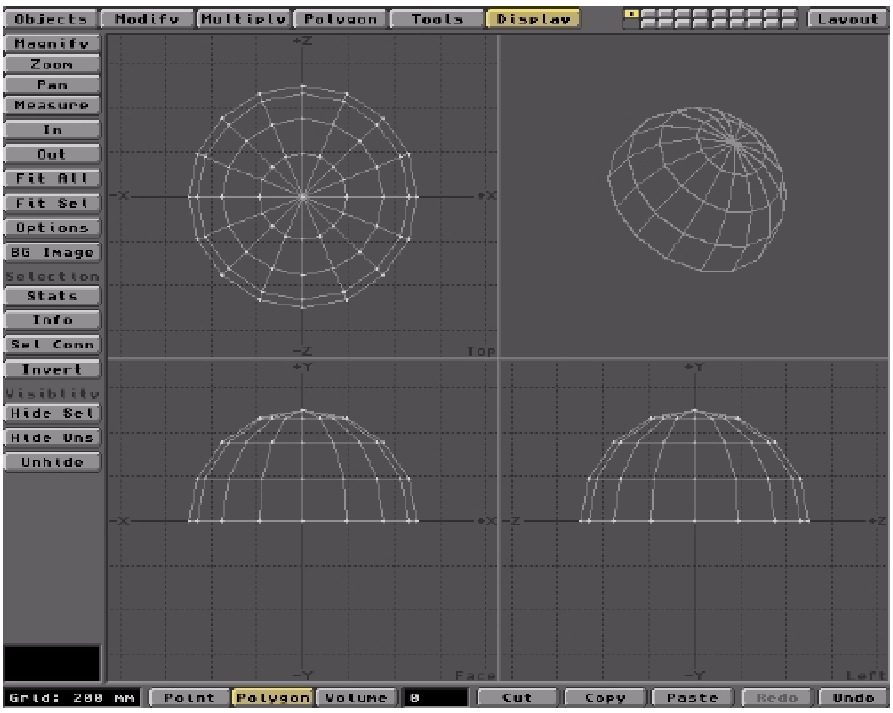
At this stage, the ball is pretty angular and faceted. Don't worry about it. The important thing is that it's composed of exactly the right number/shape of polygons (16x8=128). These will eventually be coloured Red and White, in true Boing tradition. If the ball were subdivided into smaller polygons at this early stage, you'd never get the colours right. So, whatever presentation of Boing you want to create, always generate this 'minimal ball' first.

Half a Ball is Better than One!

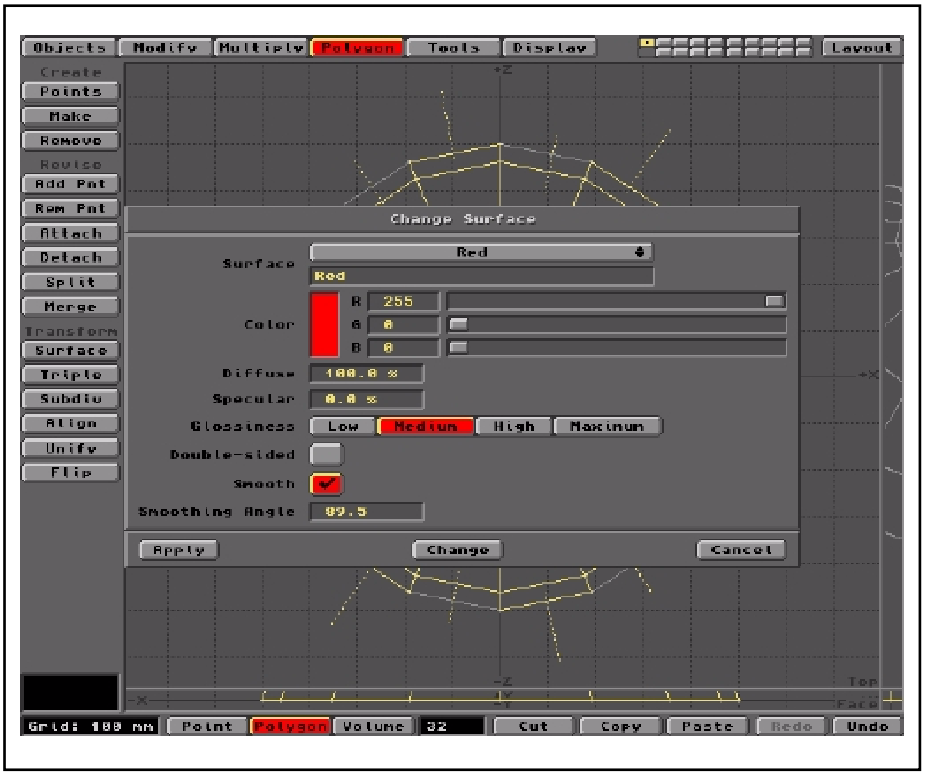
The next step may seem a bit odd. We're gonna discard half the ball and work with a hemisphere. The reason will soon become apparent if you try to select all the facets you want to be one colour, say Red. Using the three view windows, you'll find it next to impossible to determine which facet (polygon) is selected and which isn't. One hemisphere will obscure the other and you'll be pulling your hair out in no time. So, in the Face or the Left view, select all the polygons in the lower hemisphere so you can cut them out. Do this by first putting the interface into Polygon selection mode. That's the second button from the left at the bottom. Now with the LMB down, drag the cursor over all the polygons in the lower hemisphere. They'll become highlighted in yellow to indicate they're selected. You'll also see their Normals extending outward (to learn more about Normals, check out Tutorial 1).

The image on the next page shows the all the polygons in the ‘southern hemisphere’ highlighted and ready to be cut out.

OK, now click on the Cut button at the bottom of the screen and the selected polys will disappear. If everything's gone to plan, your screen should look like this...



It's worth remembering that if you missed out any polygons after releasing the LMB, you can continue with the selection process with Shift/LMB. If you continue with a straight LMB as before, the selection process is reversed! If there are still some errant polygons left after the Cut step, just select them normally and Cut them



as necessary.

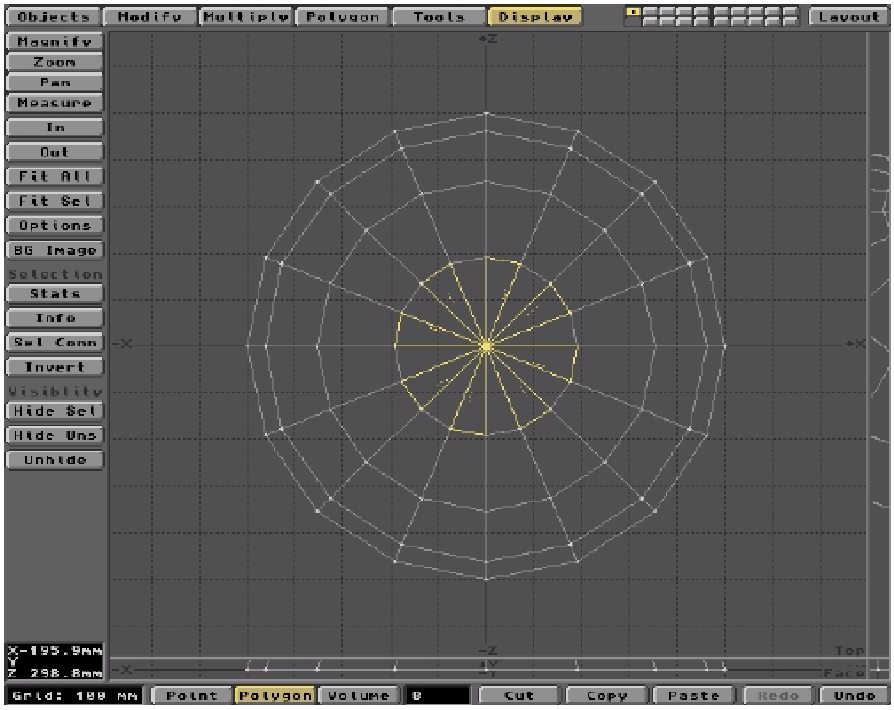
NOTE: You could just 'Hide' the ‘Selected’ hemisphere (Hide Sel) and work on one at a time. This is perfectly OK, but my method provides insight to a couple of Tools you woudn't otherwise need. As in everything LightWave, there are usually several routes to a particular goal.

Reds First

Now we have an uncluttered hemisphere, we can start selecting the facets to be made Red and those to be made White. We'll do the Reds first.

The *most important* thing to remember when selecting is to start at the ‘north pole’ and work down to the ‘equator’. I'll not explain why, but it's important. By all means experiment with alternatives and you'll soon understand my point.

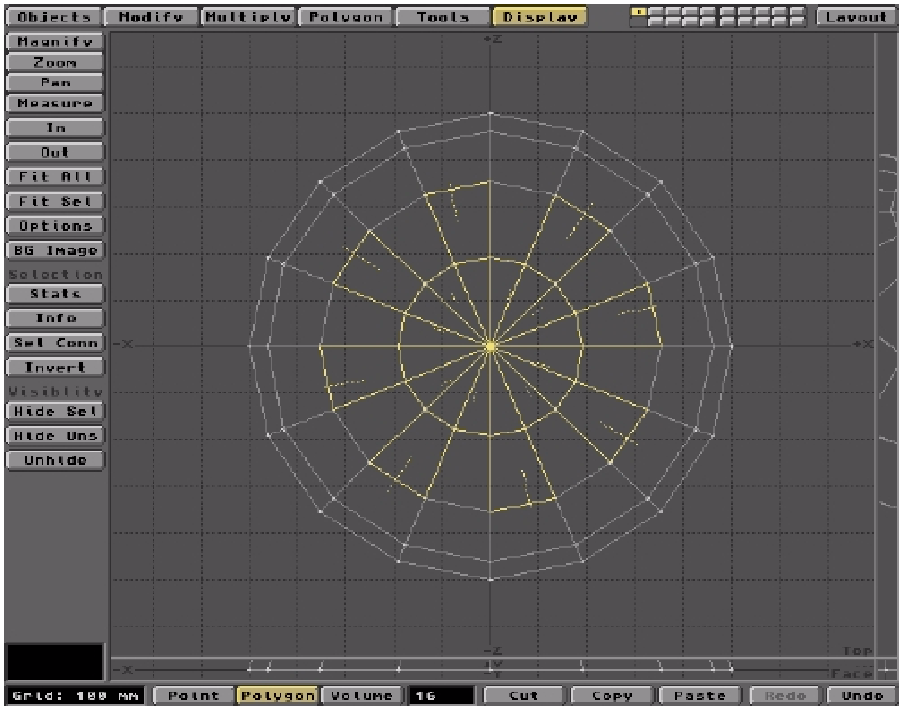
So, select one of the polar triangles using the LMB (in Polygon mode of course - check at the bottom of the screen). The adjacent polygon(s) will also be selected. That's 'cos they share a common side/edge at the position you clicked on. Selection is done by clicking on a polygon edge. Deselect the unrequired polygon(s) by reclicking these with the LMB on an edge away from the poly you actually require. It's easier to do than say. OK, continue selecting alternate polar triangles (now using Shift/LMB in order to continue selecting). Deselect (straight LMB) adjoining polys as you progress, until the required half are selected. All being well, you should see the following...



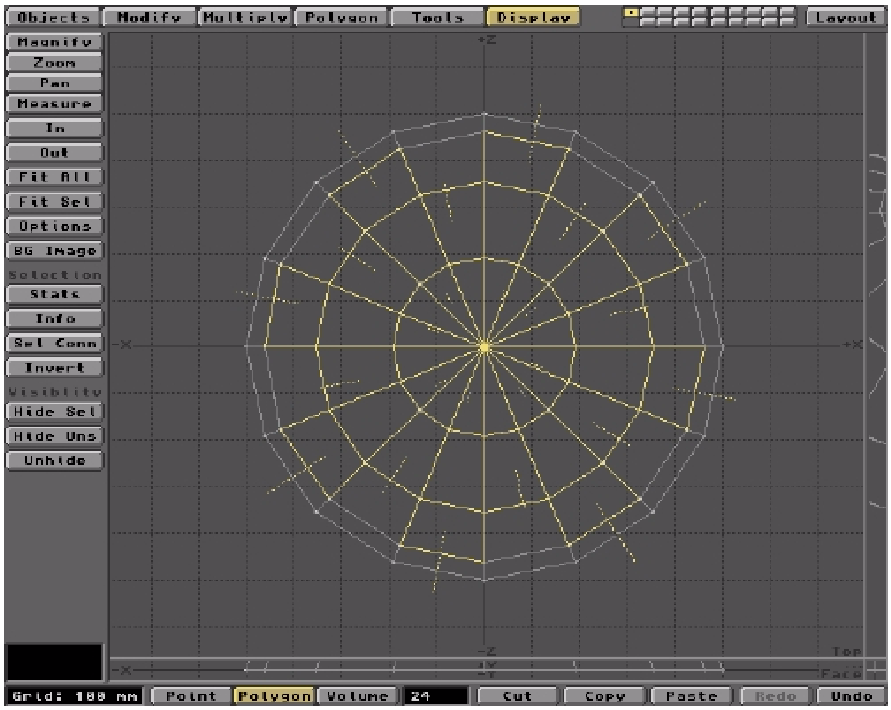
In this graphic, I've dragged the Top view window to full screen so you can more easily see the required result. Make sure only the alternate triangles are selected.

From here on in you'll need all your attention on the selecting routine. You must now look at the first ring of polys outside the polar triangles. Select the first one so it's 'between' two selected triangles. You're creating a checkerboard effect with selected polys. Deselect the adjacent poly as you progress, but take care not to deselect the required triangles! Stay away from the pole areas and stay with the outer edges. Use Shift/LMB and straight LMB to control the action. By clicking the most appropriate side of the poly you need, it becomes quite easy to keep things on track.

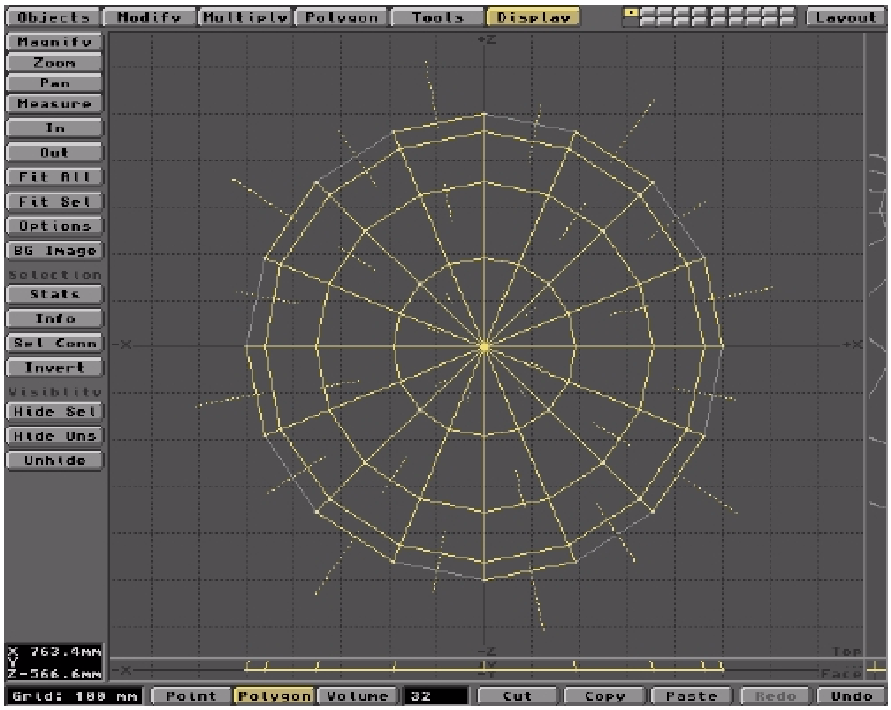
If all goes to plan, the next graphic should match your results.



You'll probably have to stare at this a while to see what's going on. The Normals are the best guide to which polys are selected and which aren't. Indeed you may already be wishing you'd gone for the simpler 12x6! Don't worry, once you get the hang of selecting the right polys, even a 24x12 will be child's play!



And the final 'equatorial' ring.



Imagine what this would look like if the other hemisphere were present with all its Reds selected...impossible!

Call 'em Red

All these selected polys will be Red in the final model, so we must tell LightWave to make 'em so. You do this by clicking on the Polygon menu button in the row across the top of the interface and then on the Surface tool in the Transform group on the left hand side. This pops up the Change Surfaces panel as shown below.

The first thing to do is change the name of the selected surface from 'Default' to something more appropriate. 'Red' sounds good to me, so type in the name on the text field. Using the RGB sliders, you can assign whatever colour you fancy for this surface. I guess (255,0,0) will be about right.

The above graphic comes from LightWave5. Earlier versions may not exhibit the colour swatches as shown here, but the process is more or less the same.

You can leave all the other Surface parameters at their default settings, though it will be useful to activate Smoothing if it's available on your panel. This will automatically invoke Phong shading when we finally get around to rendering the Surface. If Smoothing's not there, don't worry, we'll catch it later.

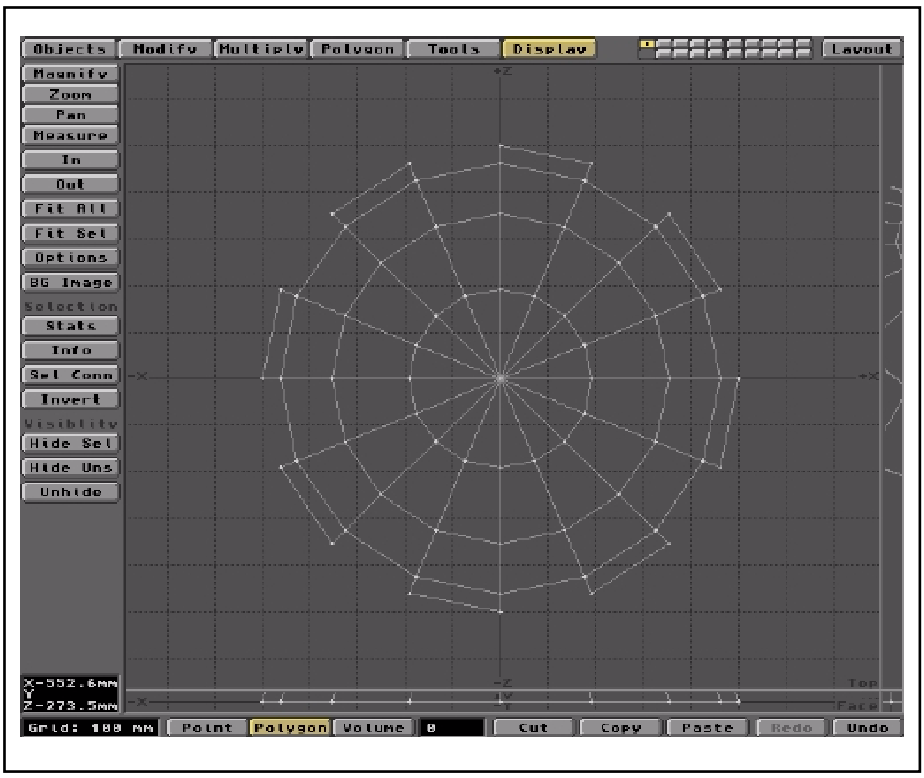
If you're interested in the Smoothing Angle parameter indicated on the panel, you can learn all about it by popping over to Tutorial 1.

Click the Apply button to fix all these settings for the named Surface (Red). The panel will close and you'll be back at the previous screen with the 'Red' polygons selected.

Hidey Ho!

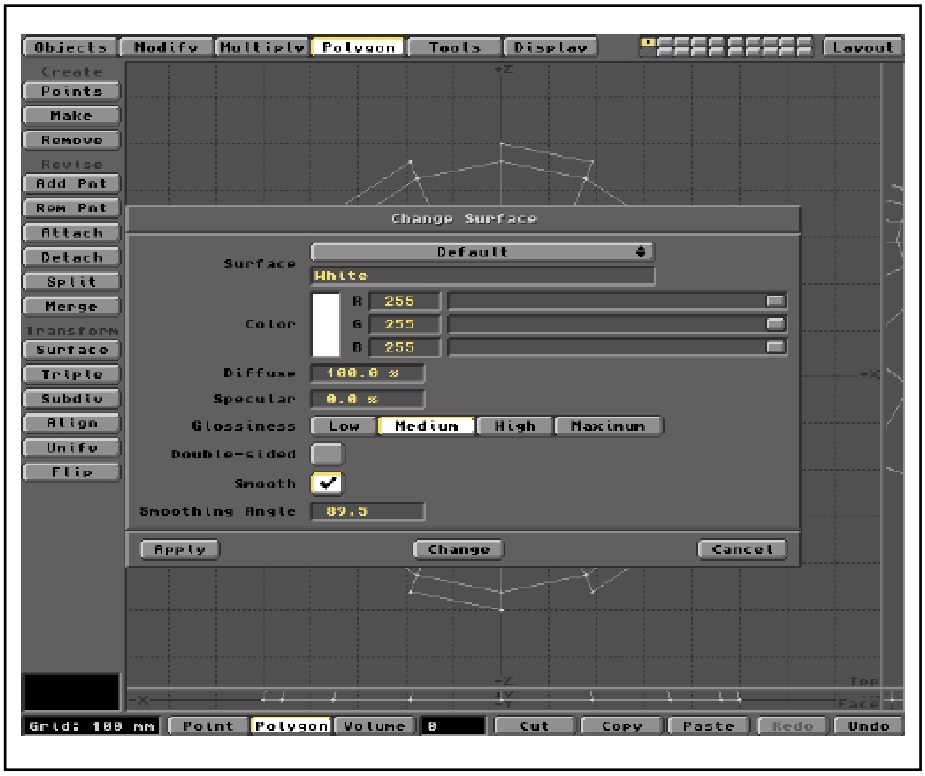
One of Modeler's most useful functions is the Hide system. You can literally hide any part of your object you'd rather not see in the view windows. You can Hide Sel(ected) items or Hide Unsel(ected) items. Afterwards, you can Unhide anything you previously hid away. Great! When something's hidden, you can apply tools to the visible parts without worrying that they'll affect what's hidden. Hidden stuff is safe from the actions you apply on screen. Let's use it to complete the Surface assignments.

Since all the Red polygons are currently selected (and what a job that was!) click the Hide Sel button. It's the first one in the Visibility group of tools at the lower left of the interface. This will result in all the Red polys vanishing from the screen. They haven't been cut, you just can't see them. And when they're hidden you can play around with the remaining stuff as you like. Here's what you get...



Now this looks pretty logical doesn't it? With the Reds hidden, the equatorial ring appears as you would expect, kinda notchy. The rest of the polys are less obvious. There should be 'holes' where all the Reds were and remaining Default polys everywhere else. The easiest way to satisfy yourself that all's well is to select everything on screen. A quick select method is to draw a lasso around the object using the RMB . All visible polygons will become selected. The Normals will confirm the picture. OK?

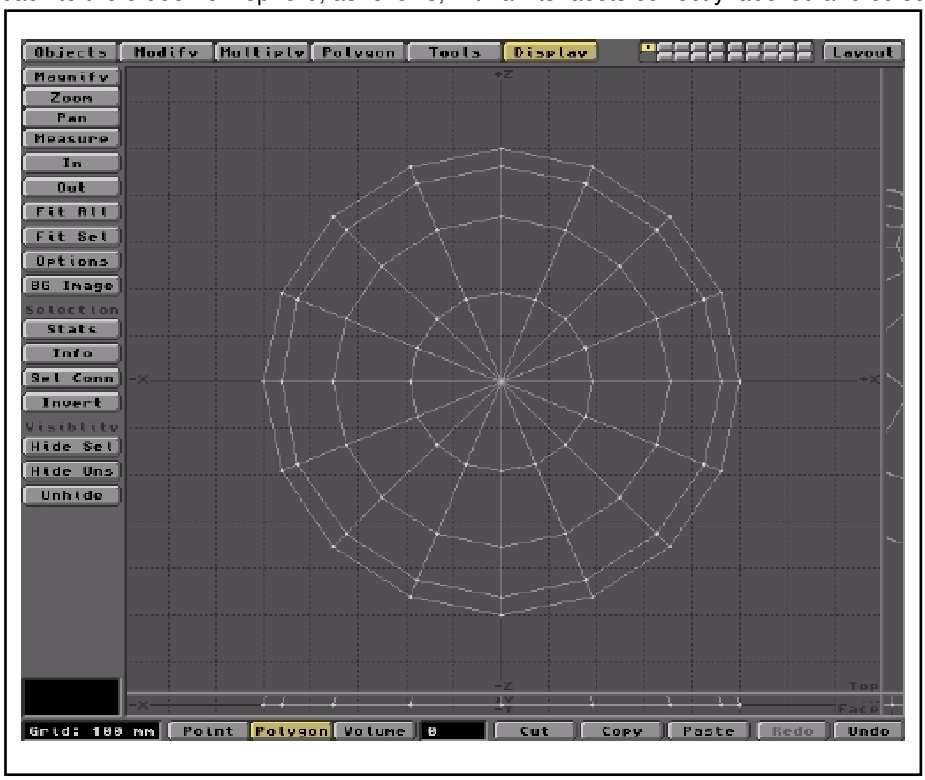
OK, whether the on-screen stuff is selected or not, it's all gonna be White, right? That's logical enough. Red is hidden so what's left has to be White. So, pop up the Change Surfaces panel (Polygon menu/Surface tool). Name the Surface White, colour it (255,255,255) and activate Smooth if available.



Here's the screen...

Click the Apply button to fix the White Surface parameters. This closes the panel and you're back to the White Polygons. Deselect them if they're selected (yellow). You can do this by going over them all with the LMB, but the quickest way to deselect Polygons (or Points for that matter) is to RMB click the appropriate Mode selection button at the bottom left of the interface.

Just to confirm everything's as it should be, Redraw the Red polys by clicking the Unhide button. You should be back to the crude hemisphere, as follows, with all its facets correctly labelled and colour assigned.



You can confirm this by checking the Polygon Stats (Statistics) panel. You get to this via Polygon mode (bottom row) using the Display menu (top right). This menu provides a Stats button in the Selection group (left side).

Click the Stats button to obtain statistical details of the polygons in the current object. The Polygon Statistics panel will confirm we've got 128 in total (16x8). It will tell us how many are three point, four point, etc. It will also confirm that all 128 polys have been assigned a Surface description. You can get details of each Surface's polys by selecting the required name (Red or White) using the scroll bar. You can select any group of polygon types (by their number of Vertices/Points). Make your selections using the 'plus' buttons. You can determine which (if any) polys are non-planar and if necessary plan to correct them. Deselect polygons with the 'minus' buttons.

OK, we've finally got one half of a crude Boing Ball. Let's create the other half. It's real easy!

No, we don't have to repeat all the previous stuff to obtain the bottom half. We simply generate a mirror image and turn it a little, geddit? Here's how.

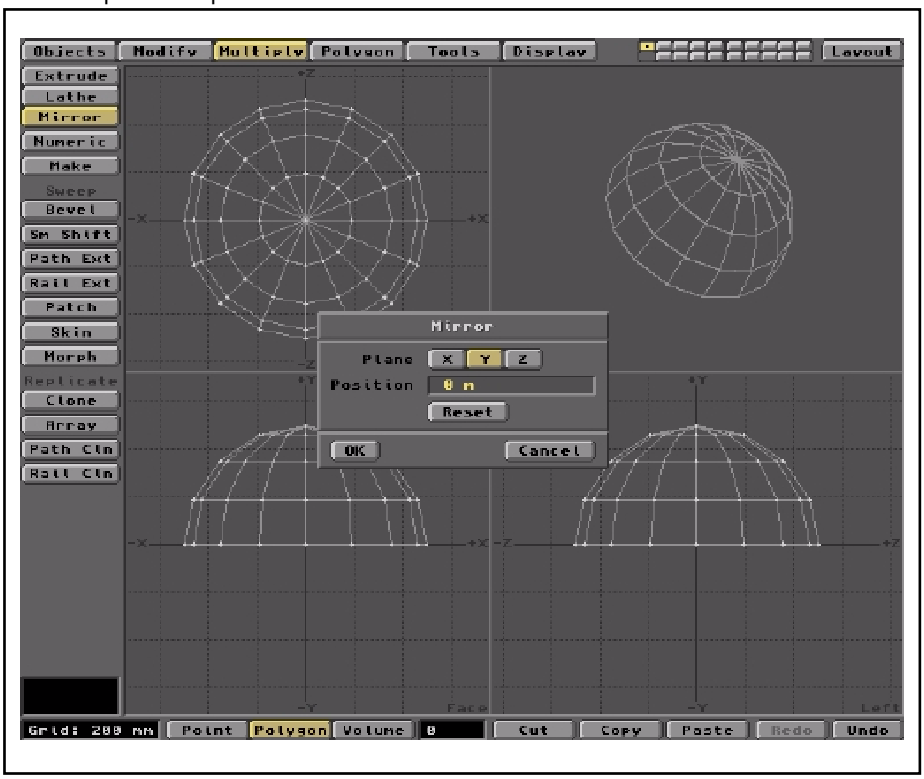
Mirror, Mirror on the Ball

Since we created the original Ball with its centre located at (0,0,0) our hemisphere is resting nicely on the XZ plane. All its 'equatorial' Points are located there. So if we use the XZ plane as a mirror, we'll get an equal and opposite hemishere, also resting on (and below) the XZ plane.

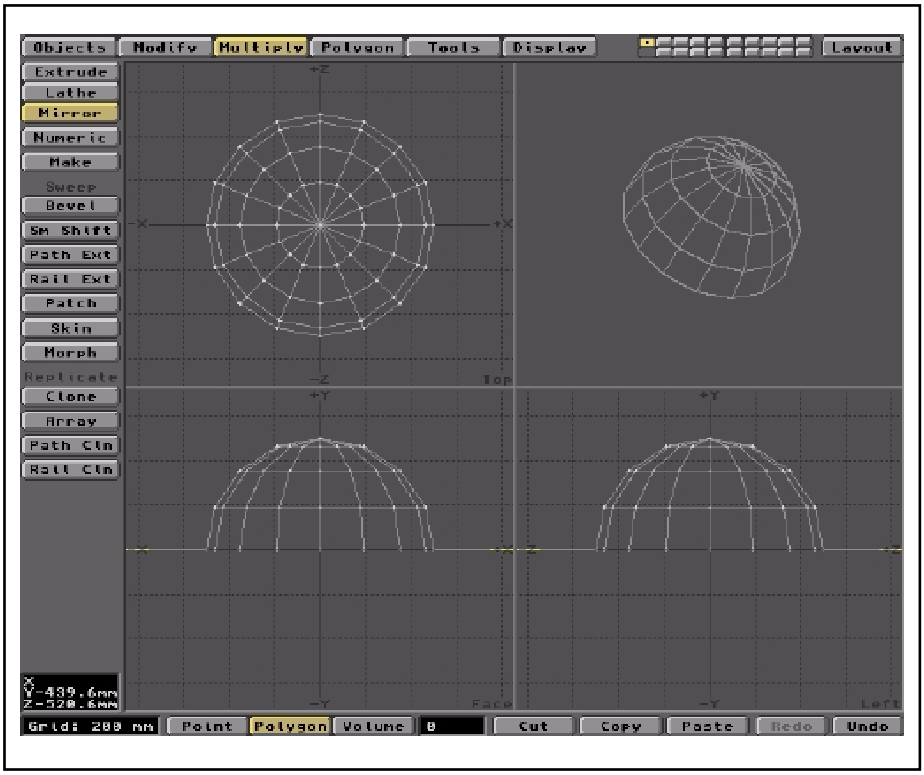
To do this, you must click on the Multiply menu (top row) and then on the Mirror button (upper left column). The cross-hairs will change to the mirror-creation cursor. A mirror plane may be placed anywhere on the view screens by clicking the cursor at the desired place. However, this can be a bit hit and miss if you wish to position the mirror very accurately (which we do). The best way is to pop up the Mirror panel by clicking on the Numeric button. This provides you with all you need to place the mirror exactly on the XZ plane.

When you first use the Mirror tool, it can be a bit confusing as to which Axis to select to set the mirror's orientation. This isn't helped much by the use of the label 'Plane' for the X,Y and Z axes. However, with only three options available, you'll quickly find it's the Y Plane/axis we need here.

Leave the Position for the Mirror at the default Zero units. This will place it in the XZ plane and coincident with the hemisphere's equator.



The Mirror's 'edge' is identified with yellow markers in the appropriate windows, as shown below.



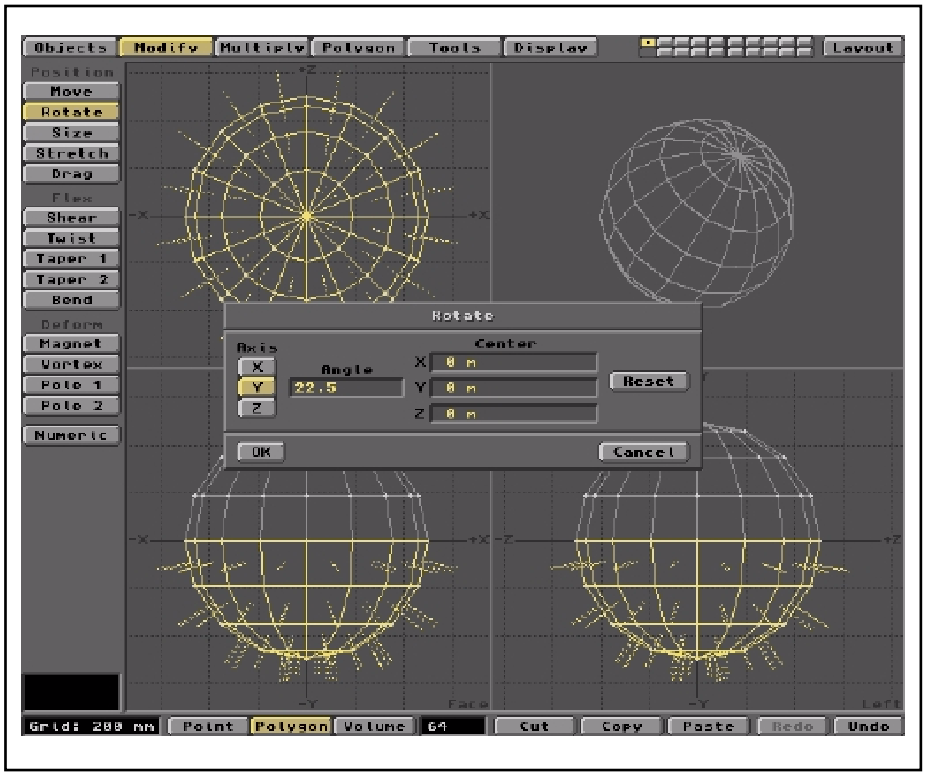
The lower edge of the hemisphere is coincident with the mirror and in this shot, mutually excluded from the graphic (dunno why).

So, to create the second hemisphere, click the Make button. An exact mirror image will be drawn.

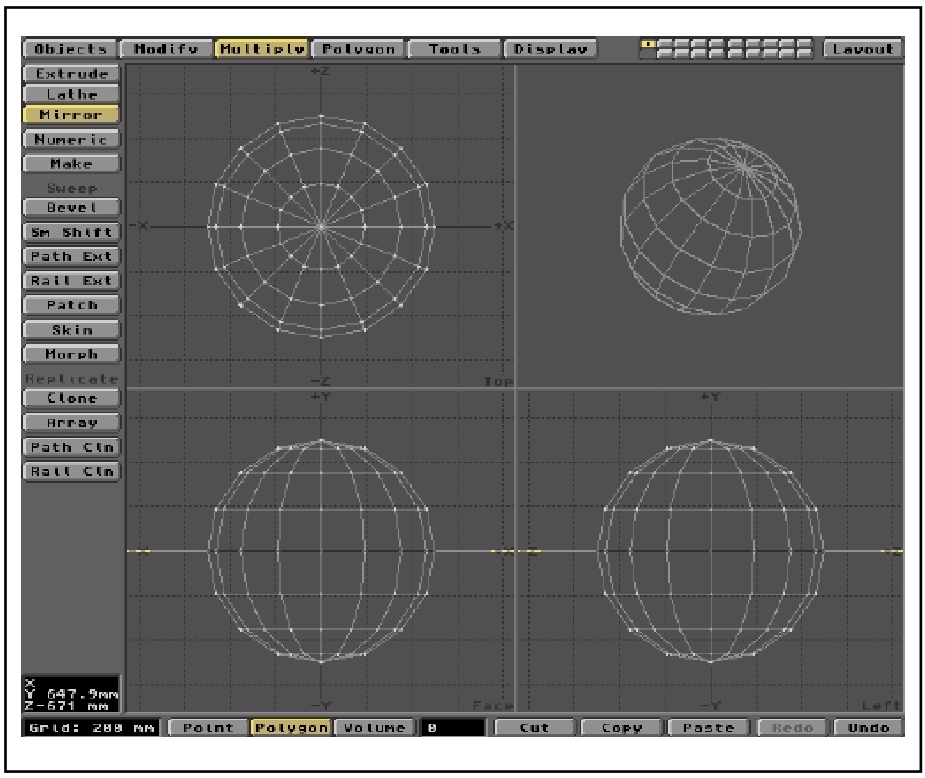
You can cancel the Mirror by clicking one of the Menu buttons (top row). The object now consists of two identical hemispheres, with Red and White surfaces in mirror image positions. That's not what we want of course. The checkerboard effect must 'cross the equator' and is easily achieved by rotating the lower hemisphere by an appropriate amount.

A Bit of Spin

The first thing to do is to select all the polys in the lower hemisphere. Use the LMB to pick them out. Next, we need to Rotate them around the axis of the sphere, that is the Y axis. Click the Modify menu (top row) and then the Rotate tool in the Position group (upper left column). The rotation must be very accurate, so click the Numeric button to pop up the Rotate parameters panel, shown below.



The required rotation is precisely one facet or 360/16 degrees. That's 22.5 degrees, so enter this value in the Y Angle field. A 12x6 Boing needs 30 degrees, a 24x12 needs 15 degrees, etc. Click OK to make the change. When it's done, the ball should appear exactly as before, though the lower colours (which you can't see of course) have been displaced by one facet.



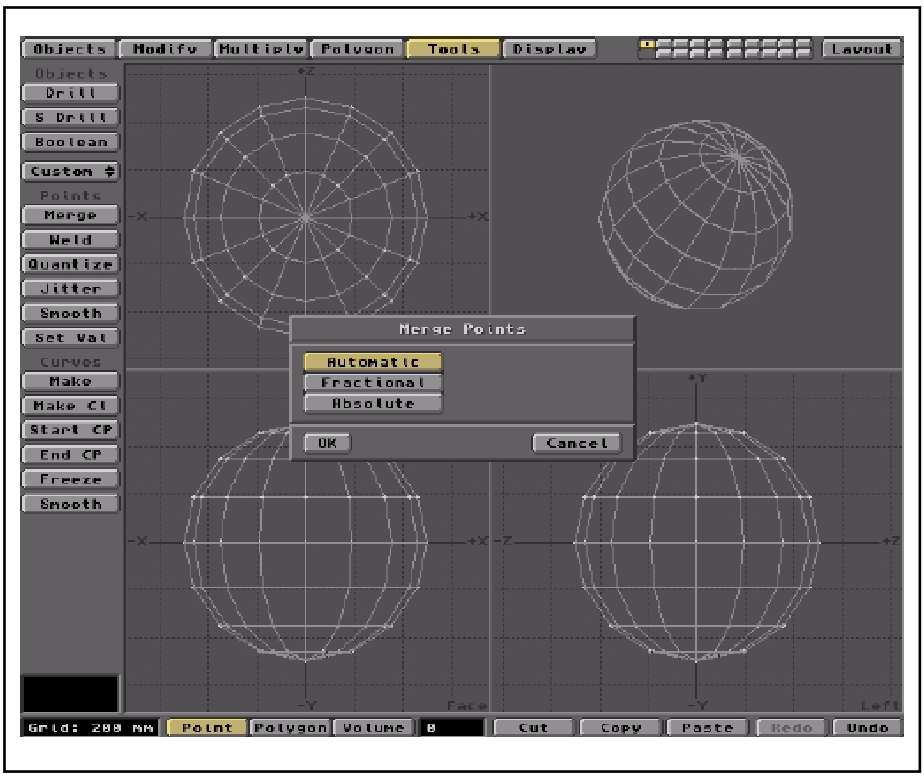
Weld or Merge Points

OK, you could consider the basic design stage is now over, but it's not quite right. The ball is actually two unattached hemispheres. The Mirror tool creates a mirror image object, but it doesn't do any joining. To fix this we need to 'weld' them together. That means dealing with the Points which define the polys, so it's done in Points selection mode (bottom row).

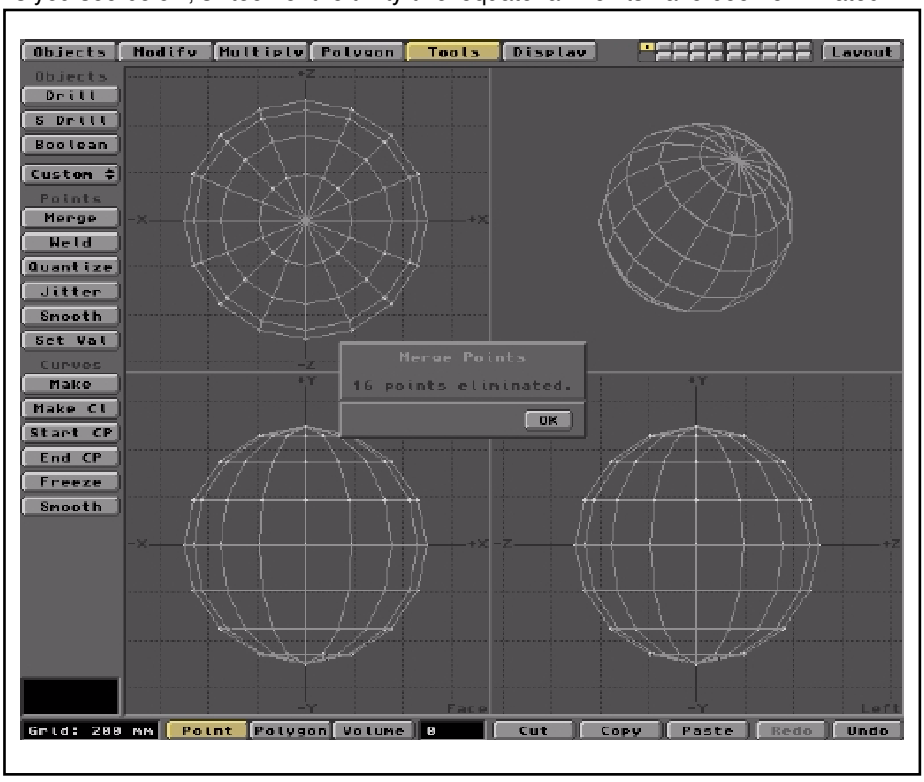
The sixteen Points around the upper 'equator' and the sixteen Points around the lower 'equator' actually lie in matching pairs. Each pair occupies the same location in 3D space. If each pair were welded together, then the hemispheres would become a single sphere. You can weld two or more Points into one using the Weld tool, but that would mean selecting each coincident pair at a time and welding them. That's OK if you like the idea. But beware, make sure you don't co-select any Points diametrically opposite the ones you're working on. Weld those as well and the ball will burst!

A far easier method is to invoke the Merge tool (Tools menu/Merge button under the Points group, centre left). This tool will automatically merge/weld together any group of Points occupying the same location. The result is a single Point made from each group. All groups are merged symultaneously and the tool works without pre-selection, according to its inbuilt parameters (point proximity, etc)

So, run a Merge Points (Automatic) on the object and it'll be right



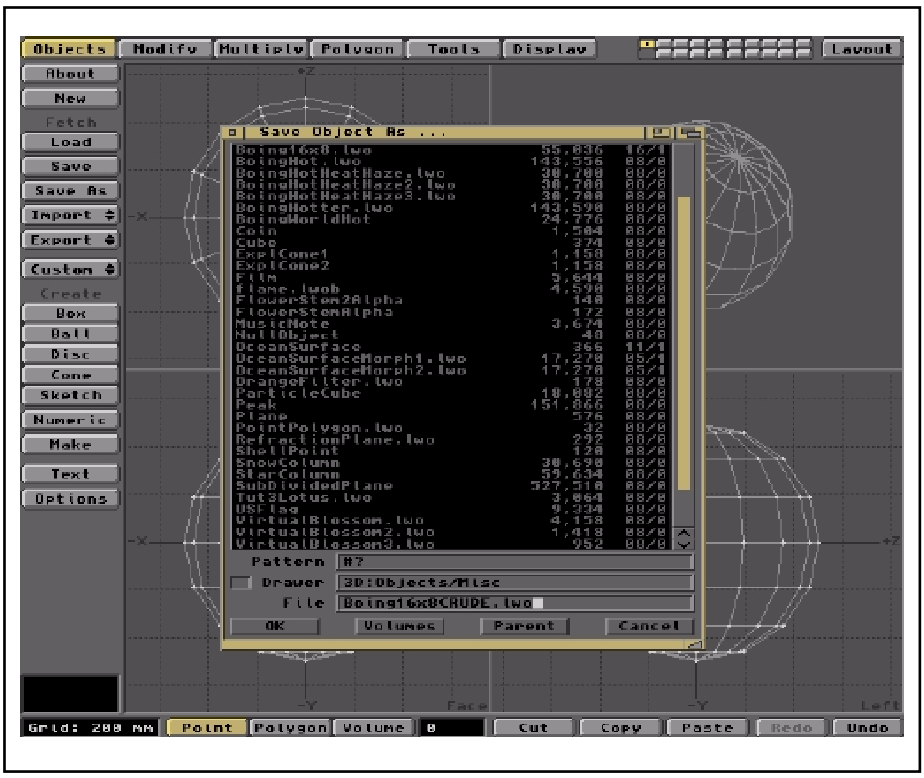
As you see below, sixteen of the thirty-two 'equatorial' Points have been eliminated. The thing is now whole.



Phew! All this for a red and white 'ball', if that's what you call it. Not very impressive! Well, this IS a blow by blow Tutorial and you'd no idea how to even start! Stay with it, the rest is the fun part!

Save it Now!

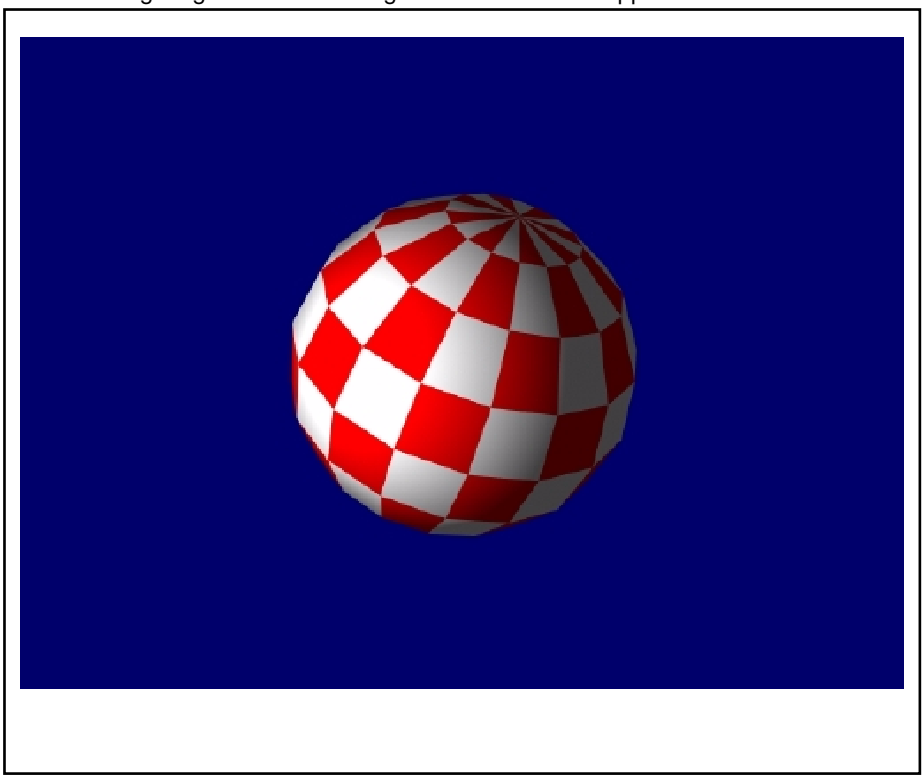
That's a heading you'll see quite often in my Tutorials and it means what it says! Get into the Saving habit and avoid a lot of grief! This crude Boing Ball, no matter how primitive it seems, involved a fair amount of work. Why lose it all for the sake of a simple Save...! So, click on the Objects menu/Save As button and get everything onto the drive. You'll be so glad you did!



Call it something descriptive. How about Boing16x8crude.lwo ? It does exactly what it says on the tin!

Shape Up!

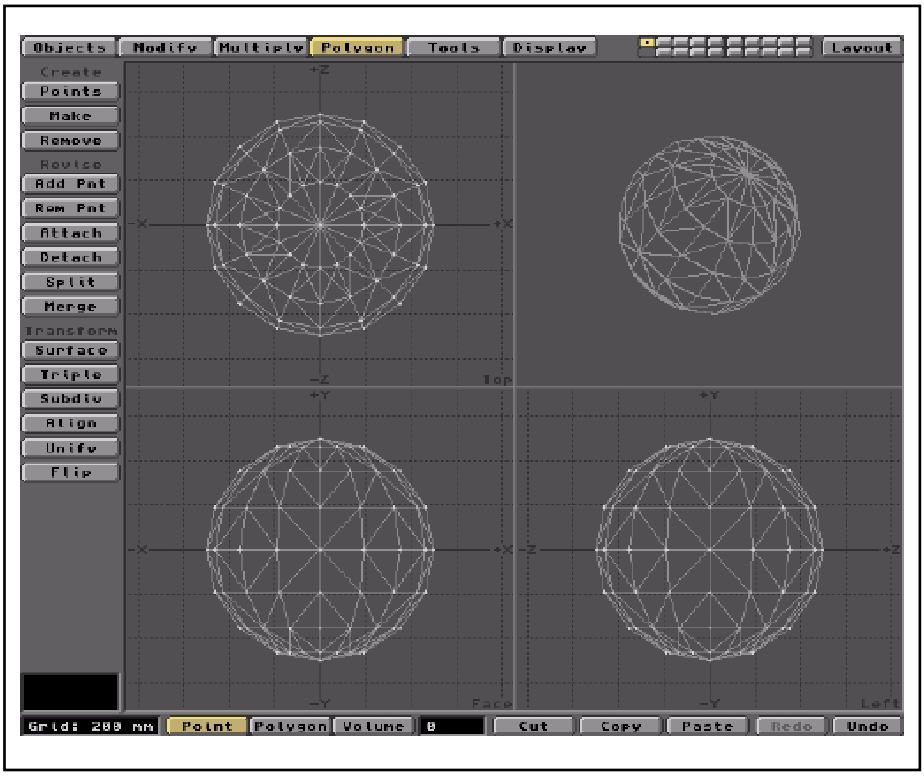
From here on, we'll be concentrating on getting the ball into shape. You might run a trial render of Boing16x8CRUDE.lwo to see how things are progressing. At least you can confirm all the Red and White bits are in their proper place. Other than that, the object really doesn't stand much scrutiny. Its polygons are too large to render really well. I tilted the axis a little using the Rotate function in Layout. The lighting uses a small setting for Ambient lighting and a second Light source off to the upper left. Check it out below.



So, the first improvement you can make is to sub-divide your polygons into smaller units. Perhaps the surest first step to good renders is to Triple all the Polygons which make up your Object. Tripling creates triangles by

sub-dividing any quad-polygons (4-sided) and any higher polys, if present in the mesh. Tripling's not essential, but it is certain. Why? 'Cos 4-sided and higher polys may not be planar (flat) and non-planar polys produce render errors. Triangles are ALWAYS planar. Think about it.

OK, let's Triple all the quad-polys in the crude Boing Ball. There are 96 of 'em, as you may have noted on the Polygon Statistics panel. It's a doddle to do with the Polygon menu/Triple tool. Here's what you get...

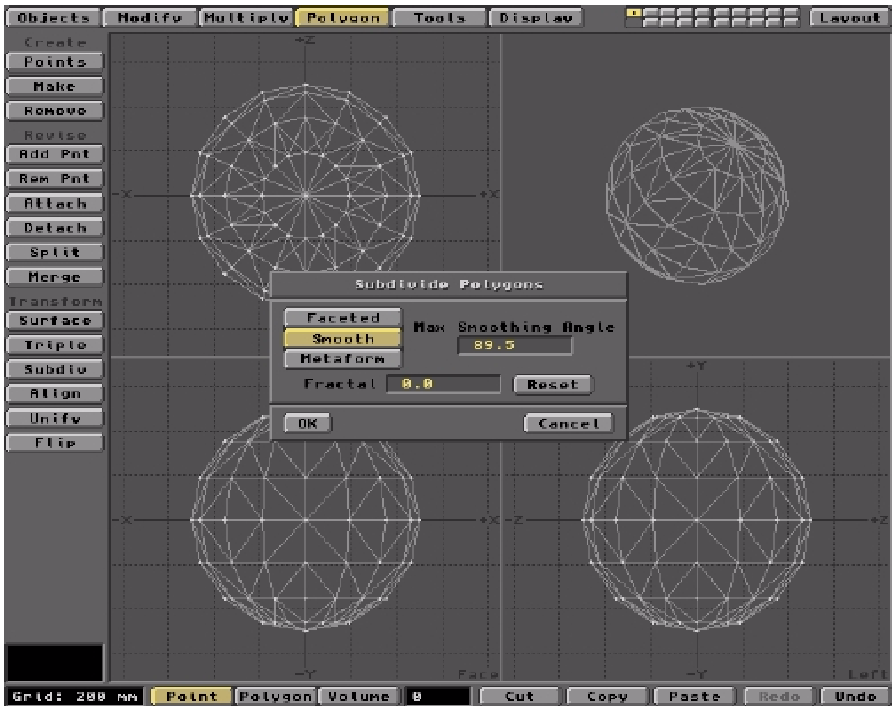


By dividing the quads into triangles, the Phong shading routine will always give you good results when you render the object. As I said, it's not always essential, but it is certain. None-the-less, we'll be creating quad-rich objects in other Tutorials which render just great. You pays your money and you takes your choice. It's usually OK, but if you hit niggling render errors, always check out the effect of Tripling your polys.

You may have noticed that the tripling process hasn't really smoothed out the angular appearance of the ball. The large flat facets have simply been divided into triangles. However, what we do have is a completely triangular mesh and that's always good. Let's look at Smoothing next.

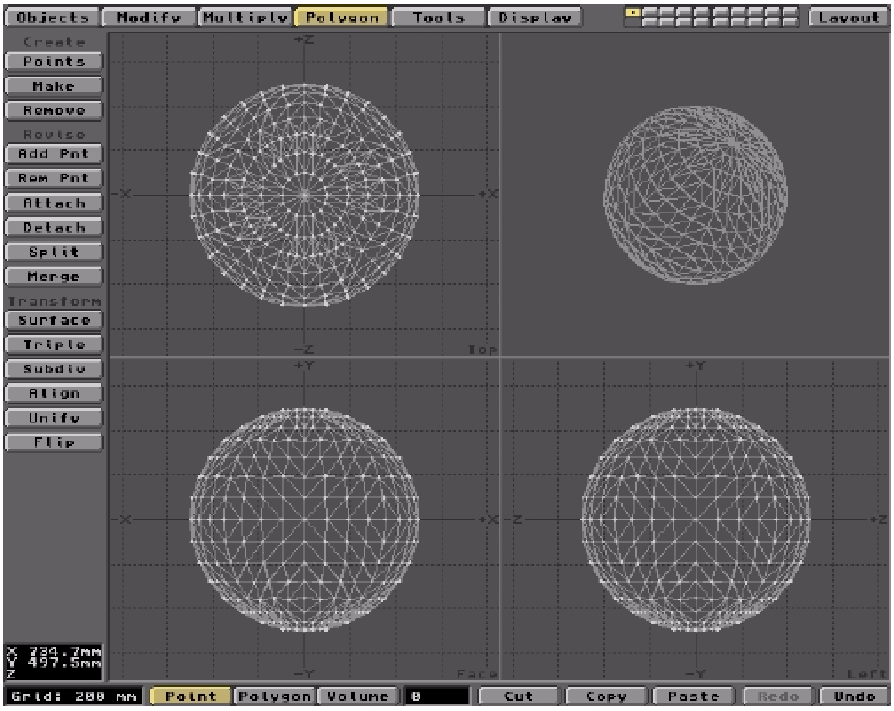
Subdivide Smooth

Subdividing requires triangles. That's the way Modeler works. If you attempt to Subdivide quad polys, you'll get an error message telling you to Triple first. Subdivide bisects the angles and edges of triangles. The Subdivide Smooth routine adds a little finesse to the process. It integrates the Smoothing Angle modifier, so you can actually smooth away the angular appearance of the mesh. New Points are inserted at elevations which impart a geometrically and therefore visually smoother surface. You invoke the process in the same manner as the Triple tool. Use the Polygon menu and click on the Subdiv button. This pops up the Subdivide Polygons panel shown below.



Here, you have several options for polygon division. Click the Smooth button to enable the smoothing algorithm. Leave the Max Smoothing Angle at the default value. This aspect is too complex to consider here, but you can learn more about the Maximum Smoothing Angle by popping over to Tutorial 1. There's a comprehensive Tutorial on the Subdivide Smooth process in the WaveGuide Manual.

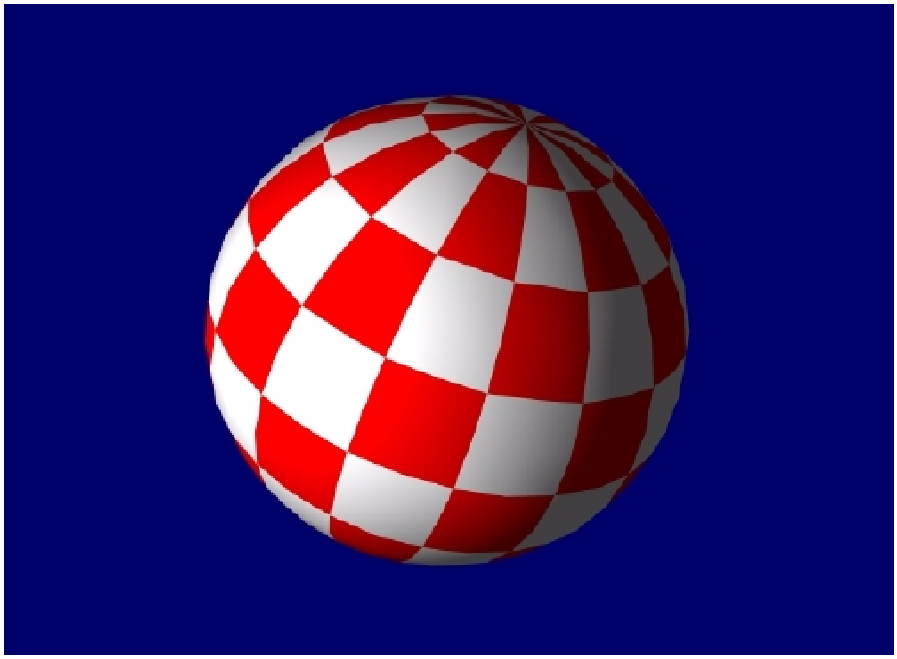
Here's what the process creates...



Notice how the curvature of the ball has been enhanced by the smoothing process.

I reckon this Boing Ball is just about finished. A further Subdiv/Smooth will make it even better, but will double the poly count and increase render times. By all means try it if you have plenty of RAM and a lot of patience. Whatever, don't forget to save the Object to the drive under an appropriate name. How about Boing16x8Final.lwo?

Here's a quick render to prove it out.



There are lots of improvements you can make to the Object in Layout. Think about making the Boing's surface slightly reflective, so you get a highlight from the lighting. Use a Spotlight to enhance the curvature and check out the Sharp Terminater feature. Use Antialiasing (Camera panel in Layout) to improve the visual quality of the image. You can animate it in classical Amiga style by setting out a Scene file in which the ball rotates about its axis. It’s only necessay to make the minimum size anim file and loop it. It only requires this particular ball to rotate 45 degrees (i.e. two segments or 2 x 22.5 degrees) to bring its position back to the starting point. Other Boing formats need different amounts. Think about it. Also think about a.... hang on, that's another Tutorial already!

Tutorial 13: How to Create an 'Ocean' Scene

Fractal Bumps, Texture Velocity, Reflection, Refraction, Lighting and a bit of Trial and Error

First - Gather Your Thoughts

The picture entitled 'Ocean' on the WaveGuide site's Gallery page is actually a still from an animation file. The animation is based on a scene I created in a few minutes of trial and error. It's something anybody can do, even a LightWave starter or wannabe! However, version 3.5 only allows a single Surface attribute to be employed, whereas in mine the ocean's surface has two. The difference is small, so if you don't have LightWave4 or 5 don't assume you are left out of it! Just use the single attribute described below and your ocean will look just as wet as the one you see here!

I used to see those fabulous LightWave images in Amiga magazines and marvel at the sheer competence of the people who made them. I was convinced that they had lots of 'inside information' about the manipulation of Lightwave 3D that I would never have. Well, after working with it for a few years it became pretty clear to me that actually, a great deal of trial and error is involved in getting those effects exactly right. In other words, anybody can do it, provided you are prepared to work at it. And it doesn't actually take that long to arrive at a pretty satisfactory result. Let me show you how.

The 'Ocean' scene is a typical case. I thought it would be interesting to attempt to capture the infinitely varying spectacle of the sea using the tools available in Lightwave's arsenal. And after all, Lightwave comes with a multitude of pre-configured Surfaces like 'Underwater' and 'Wavy This' and 'Wavy That', so it should be easy to do something like this. Frankly, I'd forget them! You really should try to generate the various surface parameters from scratch, because you'll not only learn about LightWave Surfaces, you'll create something totally unique. After all, who wants to reproduce what someone else has already made? I reckon creating unique and completely personal images is the key to LightWave's magnetism. But before you start a project like this, you should first get all the various parameters listed, mentally at least, by thinking about what's involved.

Things like the colour of the water, its transparency, and how the water shows you things below the surface. It's also reflective, so the surface you see is actually a mixture of reflected light and light seen refracted from below. You see areas of dark and light. Where the water's not too deep, you'll perceive sand at the very bottom, but also those strange darknesses under the surface. Banks of sea grass perhaps, or something more sinister? And it's all in motion. Its rippling skin, seemingly random in its complexity and yet all this movement is but part of a greater theme, typified by more regular undulations like waves and swells. So take yourself back to that fabulous beach, where the ocean danced and glistened under a setting sun. Remember how it was? Jusy dwell on it for a few moments.....OK, let's create an ocean.

Making the Parts

This Scene has only two LightWave Objects and they couldn't be much simpler. Just two large flat planes. One is used for the Ocean's surface and the other is for the sea bed. I thought we should have reasonably shallow water, so the sea bed will provide additional interest. To make the plane (for the second is merely a clone), pop into Modeler and use the Create/Box tool under the Objects menu. We don't actually require a three dimensional box as such, just a simple square or rectangle. Anyway, when you have the box icon as the mouse pointer, go into the Top view window and drag out a square/rectangle shape using the Left Mouse Button (LMB). A yellow guide frame will be created. Make it as big or as small as you fancy. We'll adjust the size of the thing when it's loaded into Layout. Since all we need is a flat plane, we don't need to give the 'box' any height, so don't drag it out in any other window. Just click the Make button to create the shape. Good, the Top view shows just four points joined together as a 'square'. The other two views show that this Object has no height, for you see just the edges view of the plane. It's a four-sided Polygon, almost the simplest shape LightWave understands.

Make 'em Triangles

You could Save the Object at this stage and it should be perfectly usable. However, you can never guarantee that Polygons with four or more sides are always flat. If any Point in the plane gets moved up or down by a minute fraction of a millimetre, it won't be flat (planar) any more and could result in rendering errors. LightWave always works best using flat Polygons and the only Polygons guaranteed flat are triangles. You can ensure all the Polygons in any Object are triangles by 'Tripling' them. So, my advice is to Triple this quadrilateral Polygon. The Triple tool will be found under the Polygon menu. Applying the Polygon/Triple routine will always give good renders. Regard this as a Rule of Thumb when creating LightWave Objects and you won't have a problem.

Name the Surfaces

The tripled plane now consists of two triangles. That's good enough. All we need to do now is give the Surface of this Object an appropriate name. Now with more complex LightWave Objects, you'll want to give different parts of the model's mesh different surface names, so different textures can be applied to them. You do this by selecting the required Polygons using the LMB in Polygon mode (the mode selectors are at the very bottom of the interface). This will highlight them in yellow. Now only yellow (selected) Polygons will receive whatever name you apply next. However, if NO Polygons are selected in this way, then LightWave assumes that ALL the Polygons present in the mesh are active (selected). So, under the Polygon menu, select the Surface button. This will pop up the Change Surface panel. Here, you'll see that the surface name is Default. This is the name LightWave applies to all the Polygons which haven't been named by you. Type in a suitable name. Let's use Ocean, because this plane will eventually become the surface of the sea. Now it should be saved to the hard disk.

Save 'em Now!

Regular saving of your Object designs will avoid a lot of grief and frustration! Even the best of programs crash (and LightWave IS the best!) so when you're working hard, just step back every fifteen mins or so and save your work. Give each update a new reference number, so all the precursors are kept as well. The file size for most meshes is pretty small and you'll be able to retrieve any earlier design stage as you want it. You'll rarely get what you want directly, so always leave your options open. The two Objects we have here are quick and easy to make, but others won't be.

So, under the Objects menu, click on Save As to pop up a filename requester. It should default to your standard LightWave Objects directory. If not, just select the path to where this Object should be saved. Call the file something like Ocean.lwo. Get into the habit of using appropriate extensions to filenames. It can help you when there are several aspects of the Object all bearing similar titles. For example, we'll be applying a procedural texture to its surface to simulate the water. This texture will be saved to the hard drive as Ocean.srf. That way, any time you need an ocean-like texture for another LightWave Object, you can apply this one.

Respray and Recycle

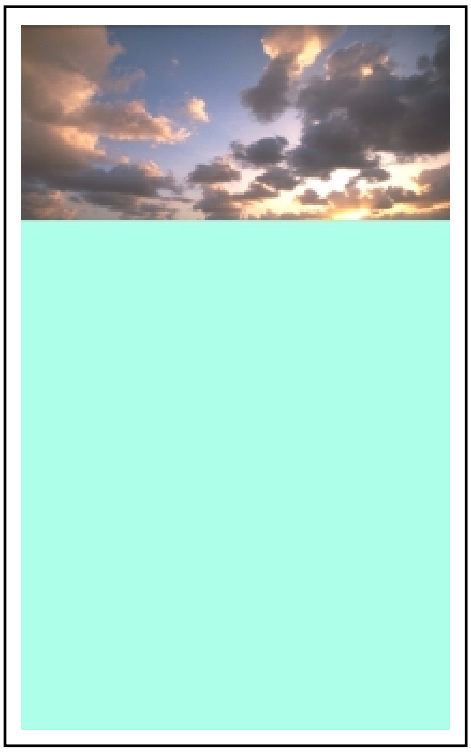
While we're still in Modeler, let's reuse this Object by saving it as the sea bed plane. Before doing so, however, go back to the Polygon/Surface pop-up panel and rename the Surface 'SeaBed'. OK, now use the Objects/Save As button and save the second Object to the drive as SeaBed.lwo. OK, the next step involves some graphics manipulation.

Some Graphics Manipulation

The third requirement for our Ocean scene is a suitable background image. Almost any 'sky' image will do, but if you can find one which has a high degree of perspective, so much the better. A setting sun is a bonus! The image I selected came from a CD of license-free images, many of which are quite superb. You see it below.

The original Image

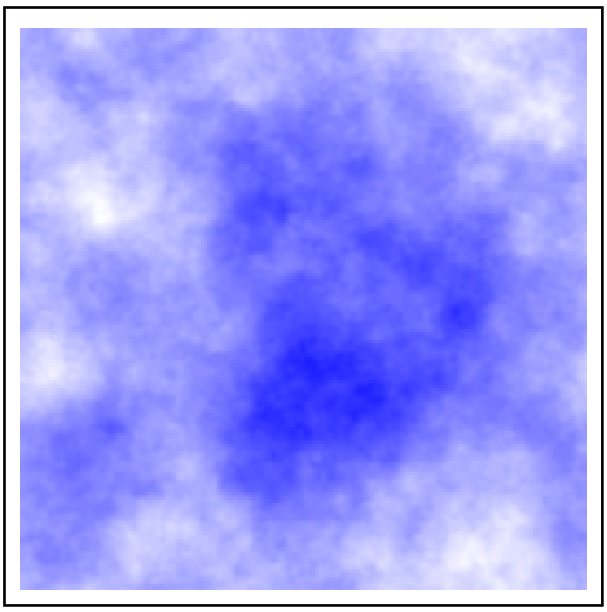
In the Ocean scene, I felt the horizon ought to be somewhere near the top of the picture. Let's say about a quarter down from the top of the frame. The horizon in this image is much lower than I wanted, but it's a great shot! So, I loaded the pic into a processing package (Photogenics) and cropped it from the top to a nominal horizon line. I then pasted this part onto the top of a tall blue page and saved the whole lot as an IFF file called OceanBackground.iff. The colour of the blank area can be anything you like. It would make sense to save this in LightWave's Images directory. Here's what it looks like after processing.



The modified Image

When this new image is used as a Background in Layout, the image's horizon is located more or less where I wanted it, much nearer the top of the frame. Remember, that the Background Image is keyed to the very top and the very bottom of the scene's frame, irrespective of their sizes. The image will be stretched or flattened to fit. I just added some redundant space to fill out the lower three-quarters of the frame. The area 'below' the new horizon line will be covered by the ocean Object, so it doesn't need anything in the way of detail.

Before we get into the mechanics of setting up the Scene, there's another Image required to get maximum impact. This is a picture suitable for use as a Reflection Image on the ocean surface. I found the original sky picture was not blue enough and gave the water a dull tone. What's best is a simple 'blue sky' picture with some white clouds. Here's the one I used.



A 'Sky' Reflection Image

You could easily create a suitable image yourself using a paint package like Photogenics. I used a pd programme called 'Clouds' to make this one. The image was taken from my old DPaint anim poking a bit of fun at Windows95-cum-98. You may remember when Bill Gates made his high profile presentation of Windows98, it crashed! Check it out on the WaveGuide website.

Anyway, none of the cloud detail from this image will be seen, just patches of colour picked out by the undulating surface of the ocean. Whatever you use, call it something like SkyReflection.iff and save it in LightWave's Images directory. Although the images I've shown here are all the same width, you needn't worry about that for the ones you actually use. They can have different widths and heights. LightWave will make them fit the frame size exactly. This can lead to distortion if it has to do a lot of adjusting. Worth bearing in mind for other projects. OK, now we have all the bits, let's put 'em together.

Setting the Scene - Load the Images

Go into Layout and without altering any of the default settings, click on the Images button at the top of the interface. This opens up the Images panel. Click on the Load Image button to pop up a file requester, which usually defaults to LightWave's Images directory. Select the background image (OceanBackground.iff) you saved earlier. A greyscale thumbnail of this will appear on the panel. Next, repeat the Load Image routine and load the SkyReflection.iff. You can use this panel to load as many Images as you require, up to a thousand, depending on your available RAM. This step simply lets LightWave know what Images you'd like to use. You next have to define how they will be applied. So, close the Images panel and move along to the Effects button.

Setting the Scene - Make an Effect

You define how Images are to be used in the Scene by clicking the Effects button to gain access to the Effects Panel. The panel offers various options, depending on the version of LightWave in use. We need to set the OceanBackground.iff as the Background Image. In later versions, this facility is under the Compositing tab. In early versions, just click on the Background Image button. Either way, you'll be presented with a list of the loaded Images. Just select the appropriate one and this sets up the background. OK, close the Effects Panel and return to Layout.

Setting the Scene - Your Option

To help with the positioning of the sea surface and sea bed Objects, you can have a low resolution greyscale version of the Background Image projected behind the LightWave 'stage'. This will enable you to position the far edge of the planes in line with the new horizon line. To get the background in place, click the Options button. This opens up the Options Panel in which various aspects of the Layout screen can be tailored to your liking. Here, we want to show the background image. Early versions have a button called Show background Image, so activate that. Later versions have this button under the Layout tab. Either way, close the panel and return to Layout.

A Bit of Backgroun

Providing you're using the camera's viewpoint (View group, Camera button) the background image will appear on the stage, with the horizon line nicely towards the top of the frame. Now this is a very useful thing to have available. Not only can we now position the planes exactly right, a projected image allows you to do all sorts of cool manipulations using LightWave's Alpha Channel system. For example, the 'Thistledown' image seen on the Gallery page made heavy use of it. A six Alpha layer composite can be seen as an animation of the same name, but you'll have to download it via the Downloads page. For now though, we'll concentrate on getting the Ocean right.

Size Matters - Check out the Grid

OK, the background image is visible in the Camera view and you should be able to see the Layout reference Grid edge on. We won't really require the reference grid in this excercise, but it's useful to have around as a datum. It's especially useful for judging relative size and scale of the objects you place on the stage and the things you do with their surfaces. If you use the View/Top button, the grid will be more easily seen. The squares are a fixed number of metres in size, no matter how close or distant you are from them. You can see the grid size in the small window, bottom right of the interface. You can alter this size and the number of squares involved via the Options Panel visited earlier. The default unit is the metre/meter, though this can be changed if you prefer something else like inches or yards. If you wish, you can also remove the grid entirely by selecting the appropriate switch. You may be surprised at how small the 'Ocean' is actually go nna be as far as LightWave 'sees' it. No worries, it's all relative. Big ocean, big seabed, big waves or small ocean, small seabed, small waves. Both look the same as long as the Camera is at the right distance. The default 'size' of a LightWave Object is determined by its size relative to the background grid when saved in Modeler.

Setting the Scene - Bring on the Actors

OK, I think I understood that, so let's now load the two Objects which make up the Ocean scene. Click on the Objects button to pop up the Objects Panel. Here, click the Load Object button to pop up a file requester. It should default to the Objects directory, but if not just pick your way to the location of the stored planes, which you saved as Ocean.lwo and SeaBed.lwo. Load both Objects and close the panel.

Setting the Scene - Fill 'er Up

In the Camera view you will see the Objects edge on and coincident with the reference grid. Activate one or other by clicking on the Object button under the Edit group. The Object you loaded last will be highlighted yellow. It will appear to be a short, yellow horizontal line. That's because you are seeing it edge on in the size it was saved. We need it to fill the entire field of view, or it will render as a simple rectangle, which is not what we want. To make the two planes fill the frame, you should resize each one in turn using the Edit/Objects - Mouse/Size controls. The Mouse tools appear when you select Objects as the edit item. The Size tool lets you resize the X,Y,Z dimensions of the slected Object using the mouse. There is a Numeric option as well. Since these two planes have no Y dimension, it's a simple task to increase their X and Y sizes. So, we'll simply increase the size of each plane by dragging the mouse. You'll see it happening in real time. Make sure each plane extends well beyond the left and right edges of the frame. When it's all stretched out nicely, make this a Keyframe.

Keyframing's the Key

Whenever you alter any aspect of a Lightwave Scene, remember to lock things in that state by resetting the first frame (Frame 0) as a Keyframe. LightWave manuals will tell you that Frame 0 is always a keyframe and so it is, but its settings will be the default settings that existed when you first entered Layout and loaded in the Objects. So, unless you alter them specifically to your scene's current situation, everything will spring back to defaults as soon as you try to render or animate your work. Remember that Frame 0 is the starting point for an animation or the only frame that will usually be rendered if you wish to create a single image. LightWave defaults its 'blank' stage settings to prepare for a thirty frame animation. You may use but one, all thirty, or add thousands more. The important thing is to use Frame 0 as the starting point for the epic you're about to create. If Frame 0 isn't right, you'll get into all sorts of difficulty.

So, go down to the bottom of the interface and press the Create Key button. This pops up a panel into which you can add a new Frame number, or key the currently selected item, or keyframe all the items in the Scene. If you've adjusted several items as we have here, click the All Items button, then OK to leave. The use of Keyframes is the clue to making good animations in LightWave, but you don't necessarily need dozens. In fact the 'Ocean' scene/animation has only the one Frame 0 as a keyframe. The reason is that none of the 'actors' in the Scene will move position. Only their Surface textures will be animated. It's all gonna be an illusion!

Setting the Scene - Get 'em into Position

We now need to orientate each plane so they appear to coincide at the horizon as defined by the background image. This involves rotating them in the Pitch orientation. To do this, click the Rotate tool in the Mouse group. Using the left and/or right mouse buttons, you will readily get the feel of this manipulation. The planes will have to tilt upwards towards the back of the stage so their rear edges coincide with the horizon line. The Ocean.lwo plane should be at a slightly smaller angle than the SeaBed.lwo, so they are reasonably wide apart near the Camera. You can adjust them by switching to the View-Side and adjusting each plane with the mouse in the Rotate mode. Just ensure they are perfectly horizontal when viewed with the Camers and they meet at the horizon, otherwise you'll get the seabed rising above the waves!

A Dab of Paint, some Lights and Ready to Roll?

Hokey Dokey, the scene is more or less set. All that's left is to apply some suitable Surface textures to the two objects and to add a light or two to help with surface reflections. You may also fancy adding a flare to the sunset if there is one in your Background Image. Optical flaring is handled very nicely by LightWave, even to the extent of creating some great camera lens abberations, should such a thing take your fancy. For now, though, we'll stick to a simple star flare. You ain't Steven Spielberg! Not yet anyway!

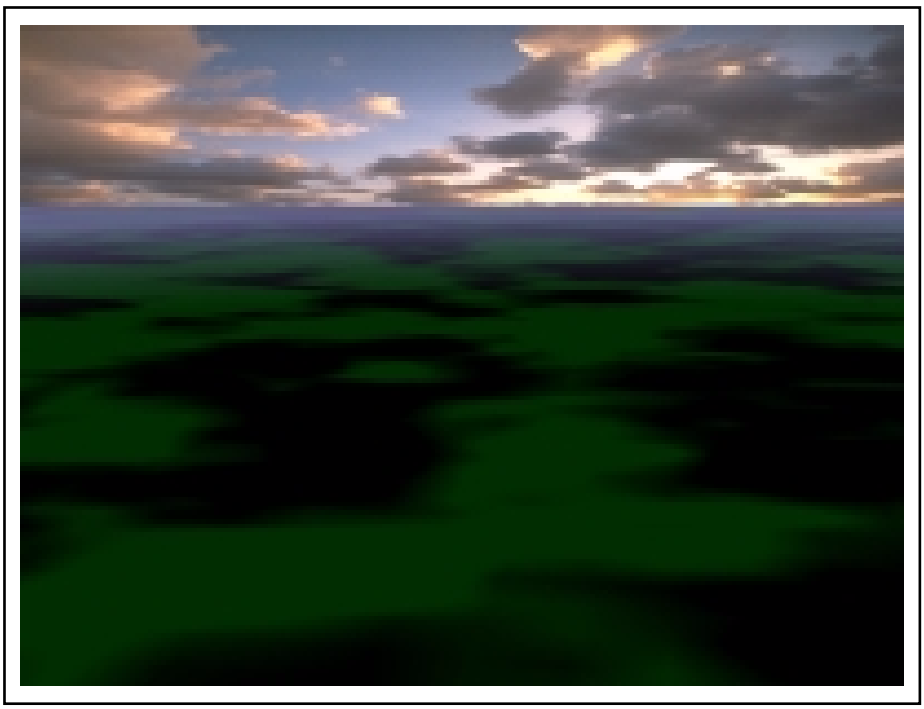
Setting the Scene - Surfacing

At the top of the interface, click on the Surfaces button. This pops up the Surfaces Panel with which you define the characteristics of all the named surfaces in the Scene. Remember, you name an object's Surface in Modeler and add colour/texture to it in Layout. A list of all the named surfaces will be presented under the scroll bar at the top of the panel. The one which sits on this bar is the Current Surface. The attributes of the Current Surface are those presented in the various fields on the panel. Scroll the bar to another Surface and the displayed attributes change with it. The panel allows you to load predefined attributes, which are stored in LightWave's Surfaces directory. Or you can rename the current surface, change its basic colour and assign all manner of optical qualities like reflectivity, glossiness, luminosity, refractive index, transparency, etc., etc. There are literally thousands of attributes you can build into any Surface . Indeed later versions of LightWave allow you to pile layer upon layer of different ones. Some of the attributes allow 'motion' to be built into them, so a surface may literally change before your eyes.

The SeaBed Surface

Let's start with the SeaBed.lwo first because it's the easier of the two to set up. You'll remember that the surface we need is called SeaBed, so ensure this name stands on the scroll bar as the Current Surface. Next, decide what basic colour you'd like it to have. A sandy colour perhaps, or a dark green to suggest a bank of seaweed. You allocate a colour using the Surface Color button alongside of which are the RGB values of the current setting. In later versions the button and other areas of the panel are given that colour. Clicking the button pops up a colour mixing panel in which the RGB values can be adjusted to taste. For my own render, I opted for a dark green base colour, but the results will be good using almost anything. The interesting bit comes with adding a texture

Alongside the RGB window is a 'T ' button. There are several of these on the panel and they give you access to various Texture Map routines which are built into LightWave's system. You can see what's on offer by pressing the button. Some options allow you to apply images as textures, much like wallpapering an object. Others are procedural textures, generated mathematically. The one we need here is Fractal Noise, arguably the most useful of LightWave's procedural routines. So, select Fractal Noise from the list, to pop up a control panel with which you alter the characteristics of the texture. Three aspects are relevant here, the Size, the Color and the Contrast values. The colour is easy, just set one up using the mixer panel. I decided on black, to give a bit of drama to the seabed. So when the texture's applied, you get the base green textured in black. The Contrast setting is akin to the texture's transparency. Small values gi ve hints of texture, high values make the texture colour more dense. A range of 0-5 is more than adequate. The Size of the texture determines how large my black areas are compared with the green surface as a whole. LightWave sets defaults for these various parameters and it does a reasonable job of getting an evenly distributed texture. You'll see the Size pop up panel has X, Y, Z parameters, which correspond to the three LightWave axes. The numbers you place here are in the same units as the Grid discussed earlier. The default Size setting is 1.0, which means every 'metre' of the surface will get about equal share of base and texture colour. This is where the Trial and Error element of LightWave makes its appearance, because you really can't predict what you'll get until you try a few different settings. It's also fun and after a few trial renders, you get a feel for how the texture behaves. This is particularly true of the Ocean surface. Here's my SeaBed object without the Ocean above it. It gives you a feel for what to aim for.



The SeaBed without the Ocean

The Ocean Surface

Having plumped for some settings for the SeaBed surface, let's move on to the nitty-gritty of getting the Ocean to look something like, well, the ocean. There's plenty of elements to play with here! The first is, of course, the basic colour. Easy. Just mix a nice blue-green ocean colour.

OK, next up is its Reflectivity. Water is pretty reflective, especially when light strikes it at a low angle. I set mine at around 60%. Just type this into the window or use the drag gadget to increase/decrease the value.
You'll notice another ' T ' for Texture alongside and a Reflection Options button as well. The texture button allows you to apply most of the mapping routines to the reflectivity aspect of the surface as it does for colour. I reckon a seawater surface has completely even reflectivity, so leave this button alone. Click the Reflection Options button and a panel pops up in which you can select from a range of possibilities as to how and what will be reflected off the surface if light strikes it at an appropriate angle. LightWave uses accurate physical calculation to determine what you see in the final render. If Natural laws say the surface will reflect light into the Camera at some point, then LightWave will replicate it. The optimum choice for this option is Spherical Reflection Image. The Ray Tracing options give the most mathematically accurate renders, but they take a lot longer to calculate and some would say, they aren't as pleasing.

Are my Seams Straight?

What the Spherical Reflection Image option does is place an image into the three-dimensional space around the LightWave stage. It effectively applies this image to the inside surface of a gigantic sphere that marks the outer limits of LightWave's universe. Hmm...creepy! And because the image has a left and right edge, the wrapping process makes a 'seam' where the two edges meet. If your image is of the non-repeating variety, then you can decide where the seam will be. It's normally at the 'back' of the sphere (zero degrees), so you don't often see it reflected. However, some renders may reflect the seam as a distinct line, so it needs to be changed. This is the function of the Seam Angle field. OK, here's where the SkyReflection.iff image comes in. Just put this into the Reflection Options using the scroll bar. Leave everything else at default and close the options panel.

Transparent and Glossy - just what I like!

Next up, you should click on the High button alongside the Glossiness setting. This controls how dispersed is the light at the point of reflection. A Low setting gives a broad reflection band, while Maximum makes the reflection small and intense. I reckon water is High but you may want to experiment with it. The Specular Level controls the

percentage of incident light reflected. It's an intensity setting basically and I reckon it should be set fairly high for this scene. Around 50% would seem appropriate. Now let's think about Transparency.

Sea water is pretty transparent isn't it? At least I hope the places you swim are! I reckon this should be set very high, say 70-80%. That way, we'll see more of the underwater stuff generated by the SeaBed surface. The Color Filter button activates colour filtering, amazing! What this does is let the colours below the transparent layer combine with those of the layer itself to create a truer visual result. What's next?

Smooth and Double-Sided!

Activate the Smoothing and Double Sided buttons. The first applies Phong Shading to the textures and makes for a better result. This is especially important when you have lots of polygons in the objects. It won't make much difference with the two flat planes in this scene, but what the hell, I like smoothing! You can learn a little more about this and the Maximum Smoothing Angle by working through Tutorial 1. The second button effectively makes all the Polygons in all the objects double-sided. Sometimes, polygons face the wrong way and render as 'holes' in the object. This button will guarantee success! The most interesting bit is next. The Bump Map texturing.

I Like it a Little Rough!

If you've stuck with me so far, you'll be glad to learn we're almost through! The last task, perhaps the most complex, is to set up a bump map routine which will look and animate like the real ocean. Click the Bump Map button to pop up the control panel. Here, the scroll bar will provide several options to be used as bump maps. The one we need here is Fractal Bumps because it's a very random routine, just like the surface of the ocean. Leave the Intensity setting at 100% but set the Amplitude high, say 500-700%. This should really rough it up! And set the Frequencies around 5. This determines how many different sub-routines are mingled together. The ocean is very very mingled!

Next is the Size which I found by trial and error, should be pretty small. My settings were 0.02, 0.1, 0.03 for X,Y and Z respectively. As for the Texture Center, I placed it away from the centre of the Layout grid by using -0.9, 1.0,1.0. This prevents us getting the effect all happening in one place on the screen. The center is where the bump mapping emanates from. The only other factor I have used is Texture Velocity which is only relevant if you want to create an animation. The velocity sets the speed at which the texture (which is itself three-dimensional) passes 'through' the scene. This is what generates the surafce 'motion' when animated. I have found the most satisfactory method is to move the texture vertically. Now this may seem counter-intuitive, but unless you want the surface to look like a fast flowing stream, vertical is best! This means setting the Y value to some appropriate number. The setting equates to metres per frame, so it should be fairly small or it will be total chaos. I've used -0.02 or -0.01 for the Y value and left the others at zero. This causes the texture to drift downwards through the scene. It may actually work better the other direction. Try it and see!

Ripples too!

Users of the later editions of LightWave can add more textures to the first one, adding extra realism. I used the Next Texture button to add a Ripples texture of fairly long wavelength to simulate waves. This texture is again frought by trial and error but well worth the effort getting right. Again, a fairly large Amplitude was applied, with 2 or 3 Sources. The velocity in this case was made small in the -Z direction, to make the waves seem to move towards the Camera.

Save the Surface for a Rainy Day

Although Surface parameters are saved within the Scene file, you may want to use this elsewhere in your artwork. To make this available quickly you can save the surface as a separate file. Just open up the Surfaces menu again and click on the Save Surface button on the panel. This pops up a file requester into which you should type a suitable name. As indicated at the start of this session, OceanSurface.srf would seem appropriate. OK this and the file will be saved to LightWave's default Surfaces directory. You will have spotted the Load Surface button also on the Surfaces panel. This is how you retrieve a saved Surface file to use at any time.

Satisfied!

That seems to be it! All over before you know it! The only thing to do now is to add a few extra Lights to increase the chances of catching some reflections and place a dummy sun, a Point light type located where you think the setting sun should be. Add a simple Lens Flare effect to this light (or lens reflections if you must) and it should look pretty good.

Finally, save the Scene under an appropriate name, 'Ocean.lws' perhaps? This should go automatically into LightWave's default Scenes directory. If you wish to animate the scene, first open the Scene menu at the top of the interface and change the number of frames to a hundred or more before saving the file. LightWave will gladly continue rendering the frames until it gets to the end. I'm thinking about five-hundred frames and a few seagulls in an Alpha layer. If you need an example of Alpha layer compositing, check out the 'Thistledown' animation on the WaveGuide site. This was done with by rendering one Alpha layer after another.

'Ocean'Footnote: In your version, perhaps the setting sun would be a little redder?

Tutorial 14: Displacement Mapping a Texture

Bumping Up a Surface - How to Create a Waving Flag

What is Texture Mapping?

LightWave allows you to generate surface texture and colour in several different ways. For example, the application of a Texture Map to a Surface is achieved by clicking the 'T' button alongside the Surface Color selector on the Surfaces Panel. This allows you to apply a variety of procedural textures to the selected Surface, via the pop up Textures Panel. You can also apply an image as a brush map using this panel. You can also modify the surface transparency and/or reflectivity in a mathematically controlled way. The 'Ocean' Tutorial describes the technique in detail.

Things that go Bump

At the bottom of the Surfaces Panel, there's another 'T' button to activate the Bump Mapping system. Bump Mapping further modifies the appearance of the Surface via a subtle control of lighting. It creates the appearance of a displaced surface by applying light and shade. So, while the rendered Surface appears to be rippled, rutted, veined, lumpy, creased, etc., it actually remains the same shape it was in the original Object. It's a very powerful system, but sometimes you need more.

What is Displacemant Mapping?

Sometimes, the Surface itself needs to be shifted around to create the desired effect. For example, consider a flag waving in the breeze. Let's say you wish to create this as a scene for an animation sequence. Though it's possible to use light and shade to create the appearance of waves, the flag object will render as a flat rectangle. Not very convincing. So, if you want realism, you need the real thing. You need the flag itself to be animated with a wave motion.

Make no Bones about It!

Now you could attempt this effect using Bones. LightWave's Bones system is a powerful way of displacing surface Polygons to make a new shape for the Object. There's a comprehensive Tutorial on the use of Bones in

the WaveGuide Manual. However, a situation like a waving flag needs a complete surface displacement, progressing across it in a controlled and fluid manner. That's not so easy with Bones. I've tried Bones as suggested in a waving flag Tutorial from a LightWave site and it proved exceedingly difficult. I don't recommend it. However, Displacement Mapping is real easy!

Real Bumps!

Displacement Mapping creates real bumps! Bumps are made by moving the surface polygons above or below their default position. They actually move 'up' or 'down' their Normals. For information on Polygon Normals, refer to Tutorial 1. The resulting bumps may be perfectly regular or completely random. They can be derived internally using one of LightWave's procedural routines or they can be derived from an image applied to the Surface as a Displacement Map. In that case, Polygons are displaced according to the luminance (brightness) of the image pixels 'located' within them. You can use any sort of image providing it has a reasonably wide brightness range. You can also create Displacement Maps with a paint package. A greyscale image is best, so you can control the highs and lows very easily. White areas raise the Polygons, black areas depress them. The 'Dune' image shown on the WaveGuide site's Gallery page was rendered using a Displacement Map created in a few minutes with Photogenics.

I Wanna Make Waves!

So how do you go about applying a Displacement Map? The starting point for this is the Objects Panel, which pops up when you click the Objects button on the Layout interface. Go to the centre right of the panel, where you'll see another of those 'T' buttons, labelled Displacement Map. If you click that, the Displacement Map panel pops up. Well it does if there's an Object loaded into Layout, but we don't have one yet. So let's not get ahead of ourselves. We need a flag to wave!

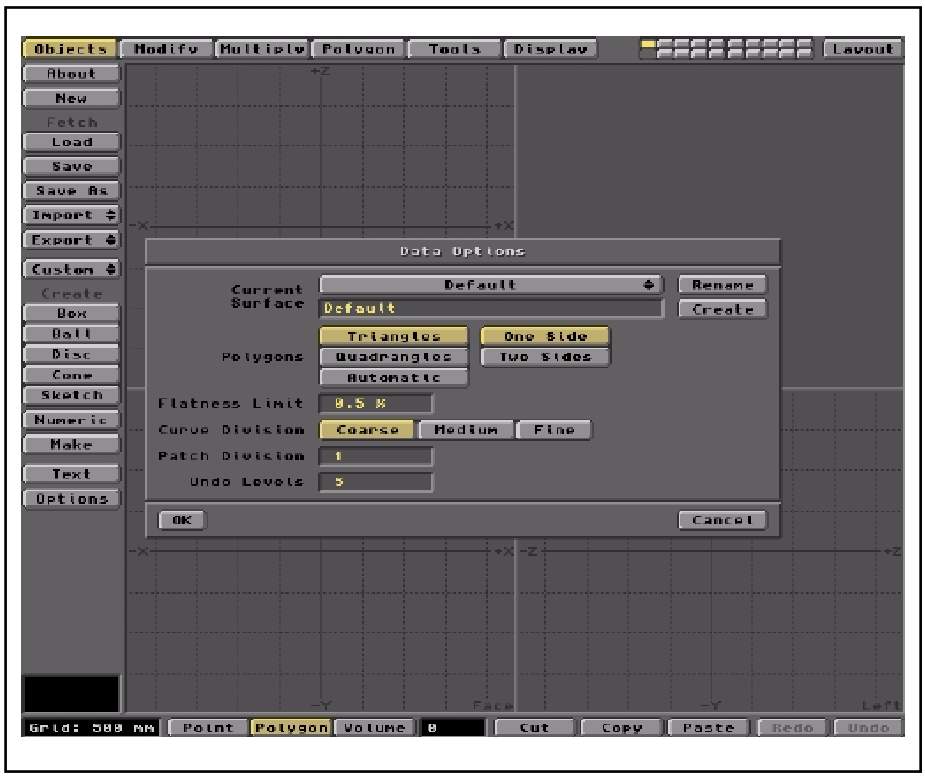
The Flag Object

OK, a flag's pretty simple isn't it? It's just a rectangular Polygon isn't it? Well, no, it's more complicated than that, especially because we require it to 'wave'. To get a really smooth, wavy surface, the polygon count should be pretty high and they ought to be triangular for optimal results. The reason why triangles are the preferred form for animated Polygons is discussed at length in the WaveGuide manual, so I'll not dwell on it here.

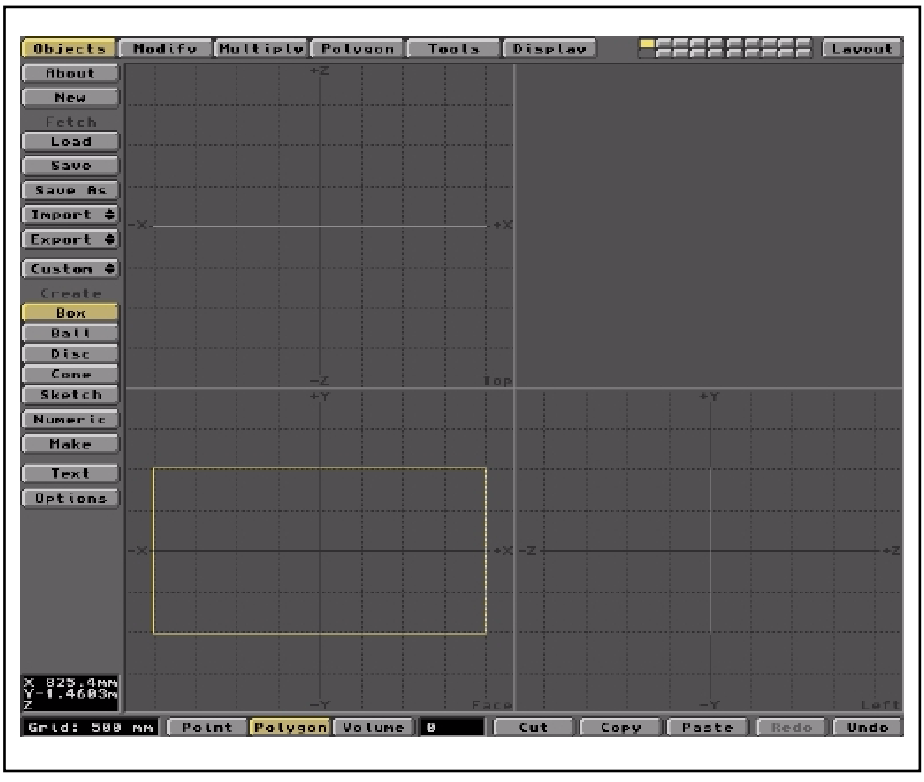
OK, pop over to Modeler and we'll create the Flag Object. There are a couple of ways to create the basic rectangular shape of the flag. You could establish (Make) Points at each corner location and then Make these into a rectangular polygon using the Polygon menu. You would do this after first selecting the Points in a clockwise (or anti-clockwise) order. If you wanted an irregularly shaped flag, using a large number of peripheral Points, this may be the best approach. For a regular, rectangular flag, the quickest method is to create a flat Box Object. The Box is a native Object format located within Modeler under the Objects/Create/Box buttons. But before we do any creating, let's make sure that the Polygons we generate are appropriate to the job in hand.

Single-Sided, Fewer Problems

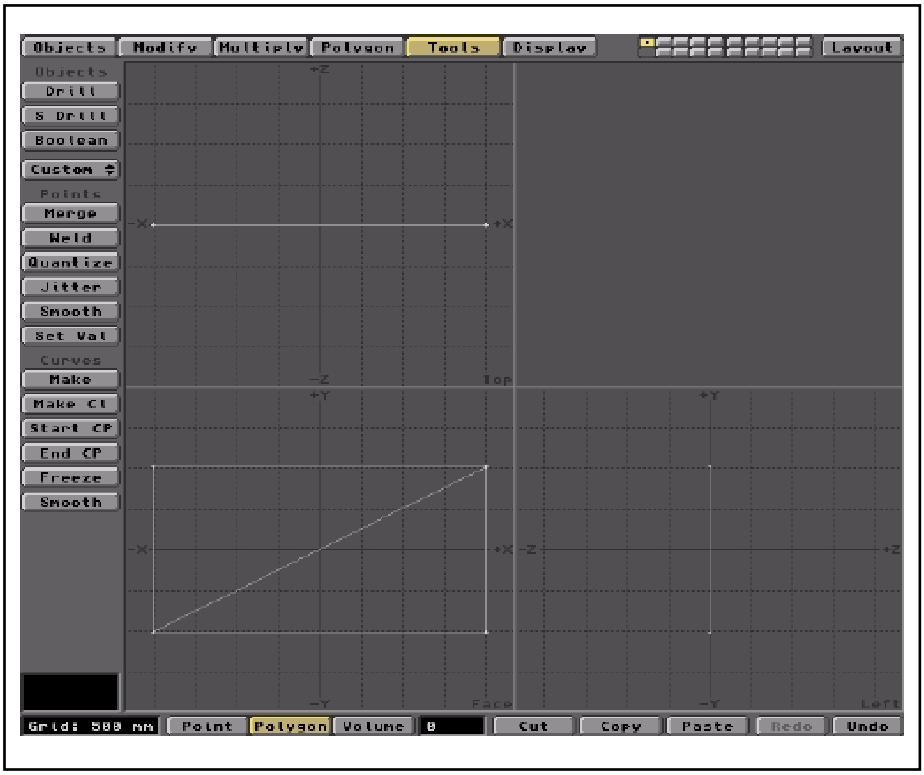
First off, click the Options button on the left hand side of the Modeler screen. This button is available after the Objects menu is selected at the top. The pop up panel allows you to control the type of polys created by your excursion into Modeler. Make 'em Single-Sided and Triangles, as shown in the following grab. You can leave all the other settings at their defaults. Using the Single-Sided setting avoids the creation of duplicate polys on the 'backside' of the Box object we'll use for the flag. It's no big deal, but if you forget that the Box object is has a front and a back face, you could end up with hundreds of polygons you don't need and that adds up to slow rendering. Triangles are best for reasons already noted.



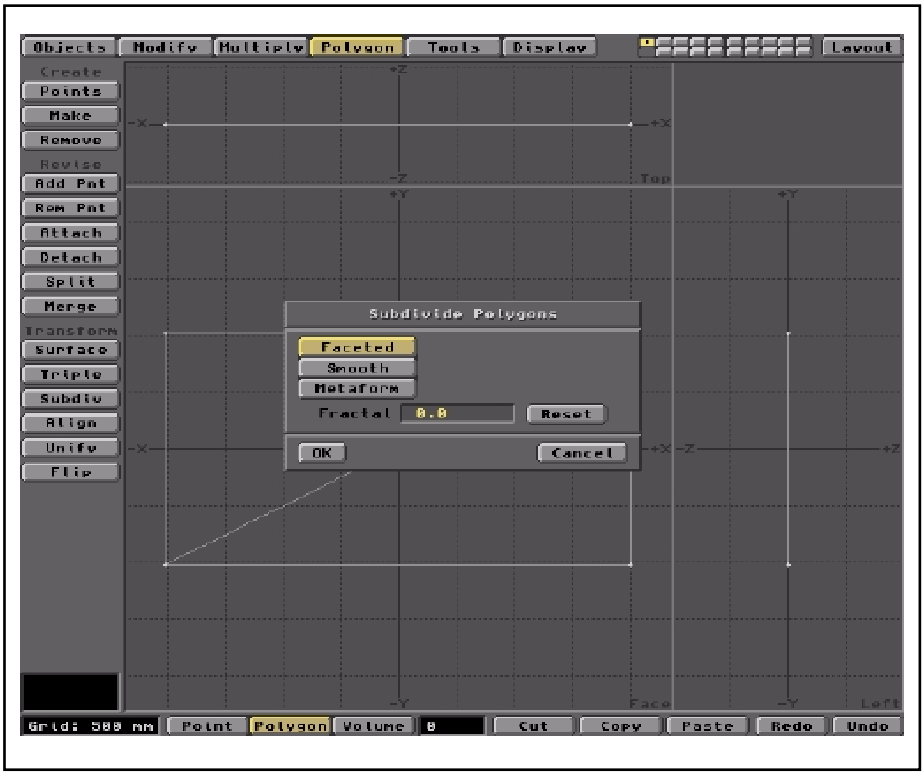
OK, now that's sorted, click the Box button. This puts the cursor into Box creation mode, so you can now draw a suitable Box outline in one or other of the View windows using the LMB. Logically, the outline of our flag should be seen in the Face window. I've dragged out a 2x1 rectangle 'cos that's the shape of the Union Jack image I intend to use on the flag surface. (There's more on the sizing aspect later.) You don't need any other dimension to the Box, so the other View windows show a simple line. The Box has no depth, it's just a flat plane. Check the following screengrab.....



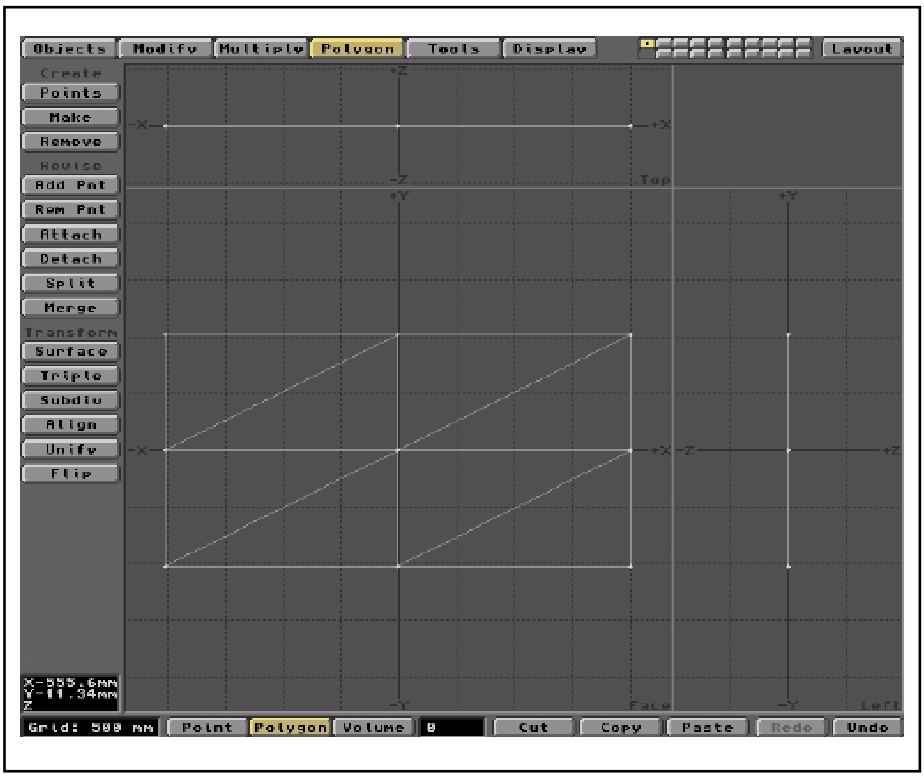
Click the Make button to complete the job. You should have something like the following screen. Remember that we specified Triangles, so the flag is split into two.



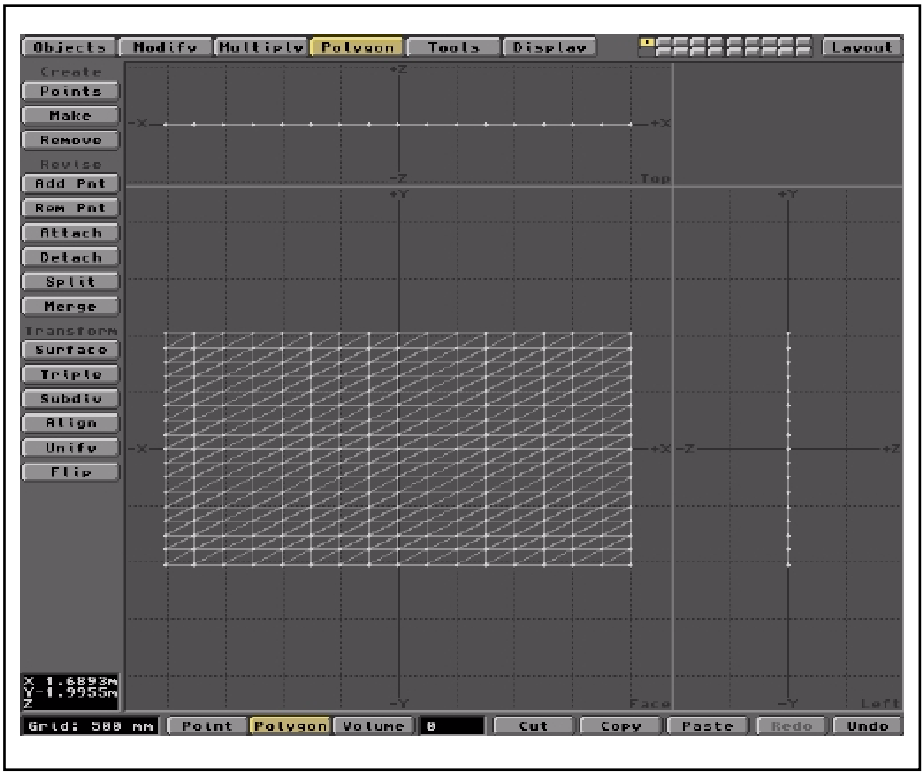
We now need to Subdivide these triangles into smaller ones. The required wave motion will not be very lifelike if the flag has only two large triangular polys to do its stuff. So, click the SubDiv button under the Polygon menu. This pops up the Subdivide Polygons panel as shown below.



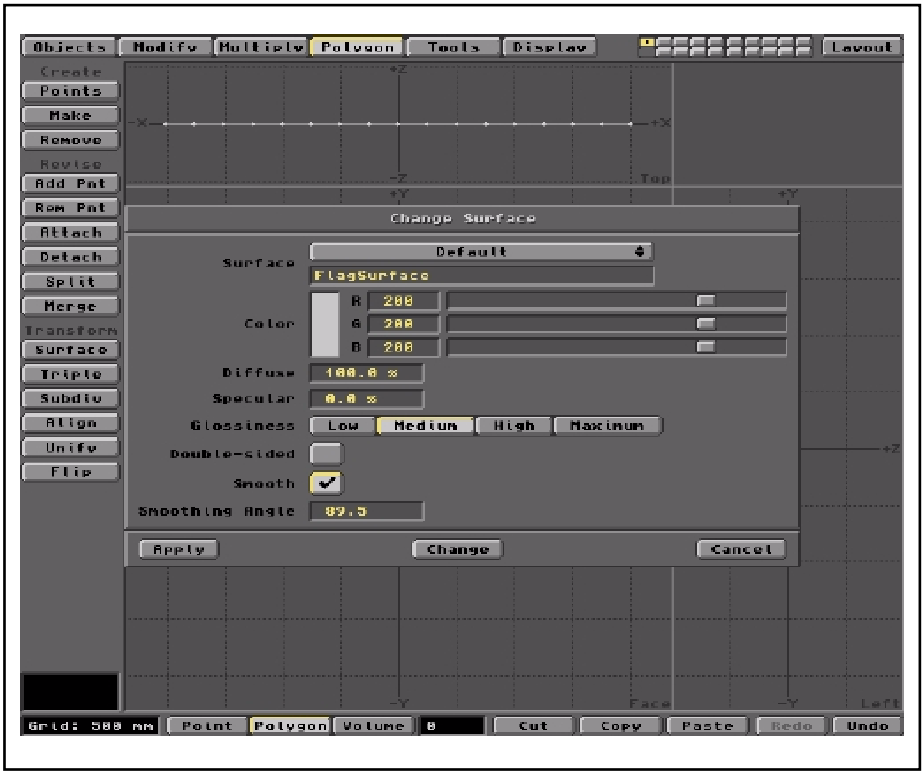
Leave all the settings at their defaults and click OK. The triangles will be divided into smaller ones. Here's what you get.



However, there are still not enough Polygons to render a smooth, wavey surface. So, Subdiv a couple more times. More if you want to achieve a really smooth result. You should get something like the folowing graphic. Here, there are 512 polygons, but you may end up with many more. If so, be sure you have plenty of RAM and a fast processor! You can check the number of Polygons using the Stats button under the Display menu.



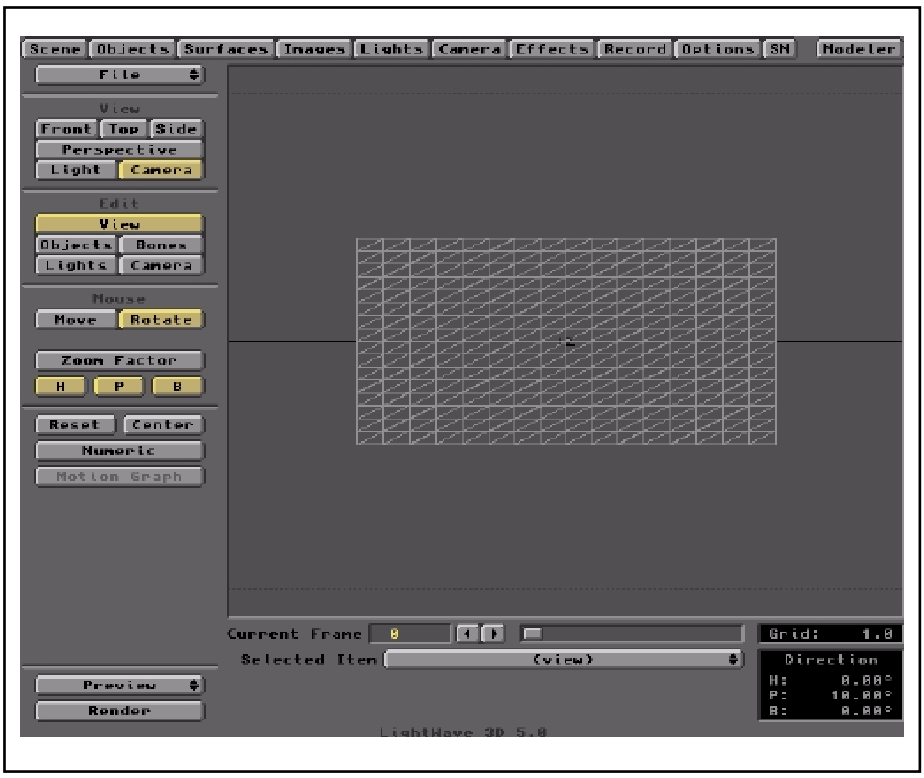
OK, I reckon this is good enough for an animation sequence. The next thing to do is to assign a suitable name to the flag's surface, so we'll recognise it when everything's transferred to Layout. Naming a Surface requires the area in question to be selected. However, since ALL our flag surface will have the same name, just go straight into he naming procedure. (The logic behind this is that LightWave considers ALL the Polygons to be selected if NONE are selected before the operation. Think about it!) You do the naming under the Polygon menu. On the left hand side, you'll see a button labelled Surface. Click this to pop up the Change Surface panel, shown below.



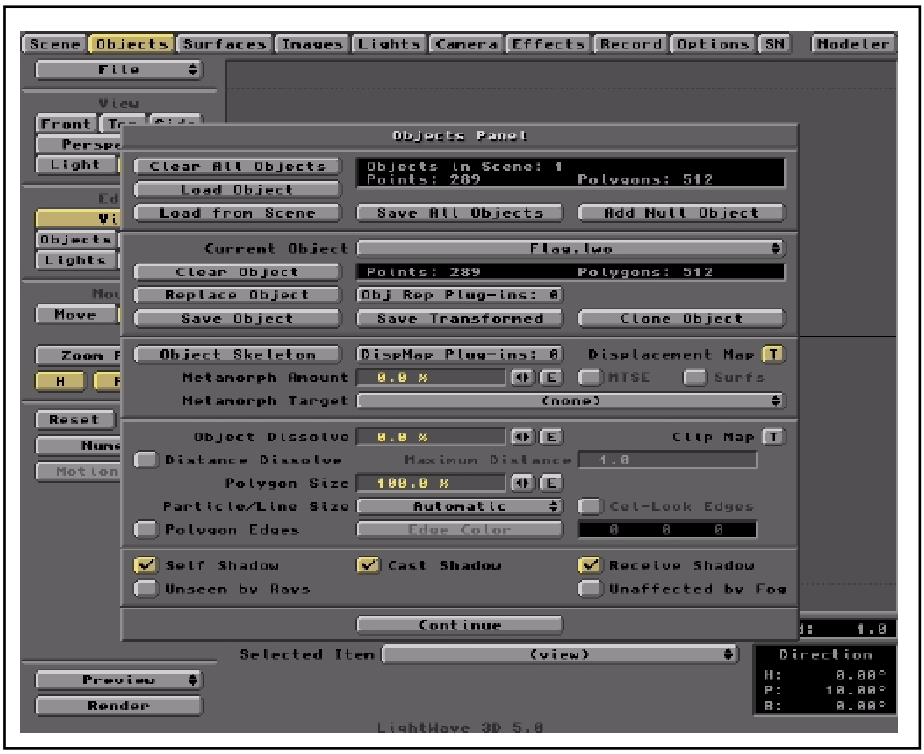
Enter an appropriate name for the surface to replace 'Default'. How about 'Flag Surface'...? You don't have to apply any particular colour settings 'cos we'll be applying an image map, which over-rides any colour values applied to the Surface. However, you should check the Smooth button to improve the render.
OK, the Flag Object is about done. All that's left is to Save it to LightWave's Objects directory under an appropriate name. I reckon 'Flag.lwo' says it all. Just click the Save button under the Objects menu and away you go. Now let's get into the interesting stuff!

Give us a Wave!

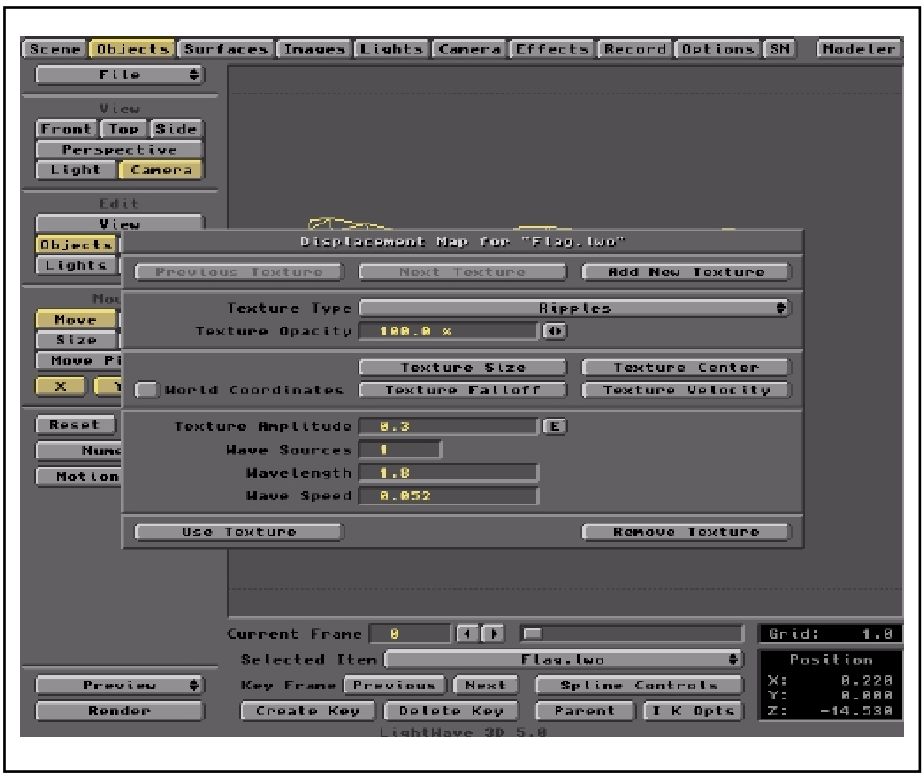
Go into Layout and load in the Flag.lwo using the Load Object button on the Objects pop up panel. It should appear somewhere near the Layout origin as shown below.



Go back into the Objects panel and look for the 'T' button labelled Displacement Map. It's on the right hand side of centre. Click the 'T' to activate the Displacement Mapping functions.



The Displacement Map panel will appear. This is where you select the procedural mapping routine you want, or nominate a Displacement Map Image. The latter will be accessible under the Planar Image Map option provided by the Texture Type scroll bar. You must have an image already loaded for LightWave to use it. You do that via the Images menu. Displacement by image mapping uses the greyscale values contained within the image to displace the object mesh. In our case, we need Ripples. That's the procedural routine to create waves across the flag surface.



The Ripples control panel has various buttons with which you can modify the effect. Let's take a look at some of these controls.

All About Ripples

I visualise the Ripples procedure as something like tossing a pebble into a pond. The point where the pebble hits the water is the Center of the texture. Everything emanates from this point. LightWave's Ripple texture is actually three-dimensional (as all procedural textures are) so the waves iradiate spherically. When applied onto a flat surface, like the pond surface or our flag, you get a series of concentric, circular waves progressing away from the splash site. The default Center position is (0,0,0) which is the axial origin. Since this is near the centre of our flag model, we'll move it off to one side to get the waving effect. Also remember that with Ripples, the wave generation is continuous. OK, that's easy enough to understand, what about the other settings?

Texture Opacity

The Opacity of a Texture is only relevant when you apply two or more different ones to the Surface. This can be done with LightWave 4 and above. The Opacity is the perentage 'contribution' any Texture has to the final surface appearance. With a single procedural Texture, leave this setting at the default 100%. OK, what about Wavelength?

Wavelength

Wavelength is the distance between the waves created by the pebble. A large rock thrown into the pond will create a large Wavelength. The waves will be big and well separated. A tiny pebble on the other hand would create smaller, more tightly spaced waves. Their Wavelength is smaller. So, Wavelength is the distance between two wave crests. That's easy enough too, but what's Amplitude?

Amplitude

Amplitude is (more or less) the wave's height above the pond surface. In the real world, a large Amplitude ripple is usually associated with a large Wavelength and vise versa. With LightWave, you can create large Wavelengths with small Amplitudes (like oceanic swells for example) and vise versa. However, if you overdo Amplitude, the waveform will become terribly messy. Start with very small settings and work upwards. You can also vary the Amplitude over time by using the Envelope ('E' button). It all adds to the fun of Ripples! But what about Wave Speed?

Wave Speed

Wave Speed is the rate at which the waves spread away from the Center. It's defined in distance units per anim frame. In real life, the speed is determined by the physical characteristics of the medium transmitting the Ripples. That is, the water, the air, the oil, the melted cheese, or whatever other gloop you can imagine. It's controlled by things like density and viscosity. In LightWave, you can make the Wave Speed whatever you like. To get a good looping animation, you'll need to spend some time on the Wave Speed element. The start and end of a loop sequence should be the same, so you don't see a 'join'. With Ripples, the wave crest at the end of the loop should be just about where the start position was.

The easy way to figure this out is to first decide how many frames you wish to use for the loop sequence. You should really set that initially, using the Scene menu. Here, you can fix the number of frames in the animation and other stuff (which are probably best left at default values). Once you've fixed the number of frames in the loop, it's time to estimate the Wave Speed. For the mathematicians out there, it's clear that the wavelength should be divided by the frame number. That way, the start and end frames will appear similar. The answer to this division is the Wave Speed. In practice, it inevitably requires a little tweeking, but that's the gist of it. OK, let's talk about the number of Sources.

Sources

Sources is a bit more complex. Imagine two or more pebbles hitting the pond surface at the same time. They all have the same Centre, Wavelength, Velocity, etc., but they transmit their waves with differing vectors. In essence, you get a mixing effect. In some places, the waves will reinforce one another, in others they negate one another. The result is a more complex pattern of resultant waves. The larger the number of Sources, the more complex (and slower rendering) is the effect. Numbers should be kept fairly low, six or less. Large numbers add nothing to the result other than slowing down the render speed. For our waving flag scene, a single Source is best.

Other Parameters

The only other parameters of note are the Texture Size, World Coordinates and Falloff buttons. The Size parameter is probably the most important of these. It's related to the overall size of the Surface being textured. The default values are usually about right. LightWave does a reasonably good job of matching the size of the Texture to the size of the Surface. However, if you wanted to make the Texture very small or very large in relation to the size of the Surface, this is where you'd do it.
If you check World Coordinates, the Texture mapping routine is 'locked' into the Layout grid coordinates, rather than the Surface's. This means that the Texture will not move 'with' the Surface as the Object moves. The Surface will appear to 'move through' the stationary texture. Wierd!

Falloff determines whether the Texture will 'fade away' with distance. It's expressed as a percentage reduction per unit of distance from the Center. This would be activated if you were trying to simulate real ripples on a pond. The wave gets fainter as the distance away from the Center increases.
OK, let's talk about flags!

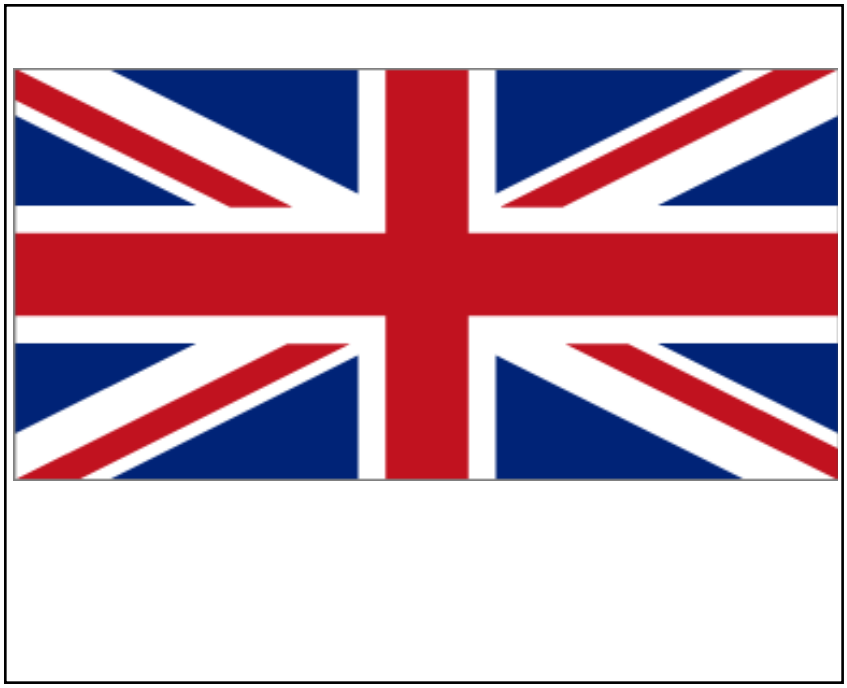
Flags of the World and Beyond

Now I've assumed we'll create a National Flag of some kind, but you may have different ideas! If you want to advertise your local club or group or the political party you favour, be my guest! Just use whatever image you'd like to have up there, waving in the breeze. All that's needed is a reasonably good image. This should be in a format recognised by LightWave, so Gifs are out I'm afraid (at least as far as the Amiga versions of LightWave are concerned).

For National Flags, a good source is the World Flag Database, easily found on the Internet.. From here, you can download whatever takes your fancy. They're all in .gif format, so Amigans will need to convert these into .iff or .jpeg format before use.

The database gives details on the height to width ratio of every flag. You'll be surprised to see how they vary! Anyway, if you want your waving flag animation to be a true representation, make sure you note the size ratio of length to width. This should then be applied to the flag mesh we made earlier. Just load it back into Modeler and use the Stretch tool (under the Modify menu) to make the ratio correct. If you don't do this, the flag image may appear too squashed or stretched out.

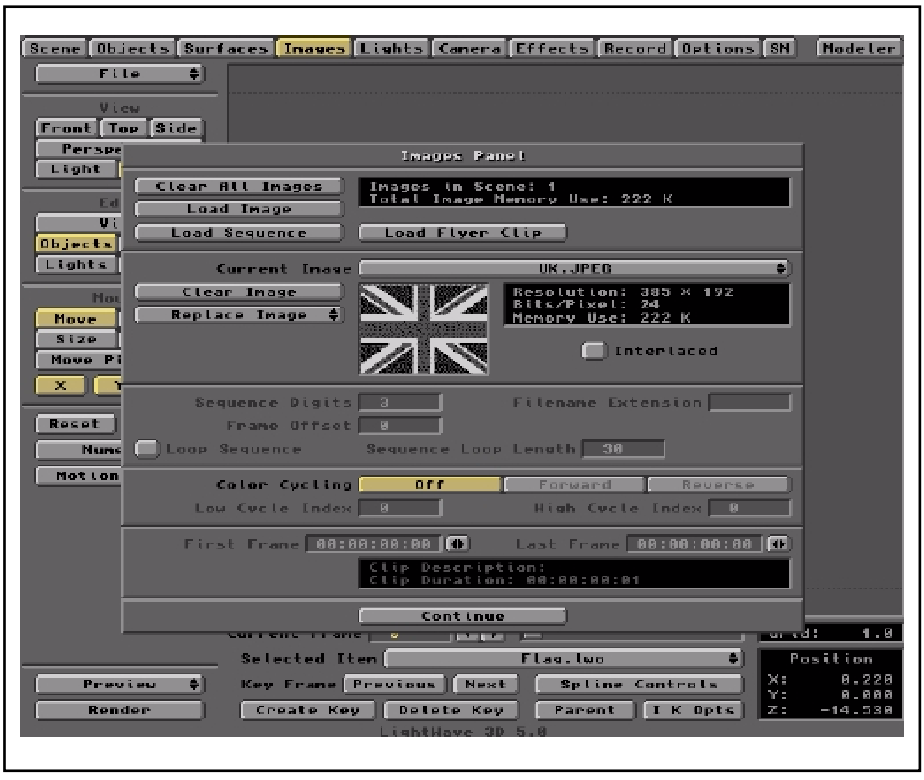
As I said at the start, I'll be using the Union Jack. This has a 2:1 size ratio (as we designed in Modeler) so the flag image will fit perfectly onto the Surface. Here's the image I downloaded from the database.



It's All Coming Together

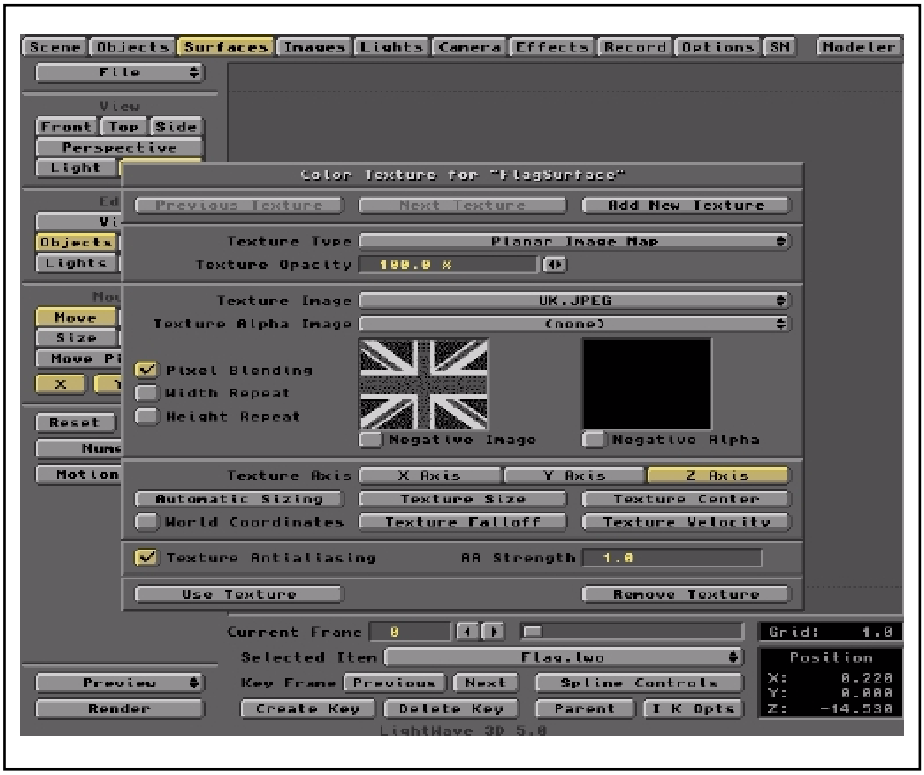
And about time too! Let's go back to Layout and load up our .iff or .jpeg image by clicking the Images menu button. This provides the Images control panel. Shown below.

Click the Load Image button to get a file requester. Here, locate the desired flag image and away ya go!



The next task is to tell LightWave to use this image as a colour map on the surface of the flag object. So, click the Surfaces menu button. This pops up the Surfaces panel, with which you can apply the image as a Planar Image Map to the 'Flag Surface' we defined earlier. Do this by clicking the 'T' button next to the colour selector. This pops the Color Texture panel for the selected Surface. As you'll see in the image below, this is being used to project the image along the Z axis. That's the direction we'll be viewing the flag, so make sure you select the correct axis for your application.

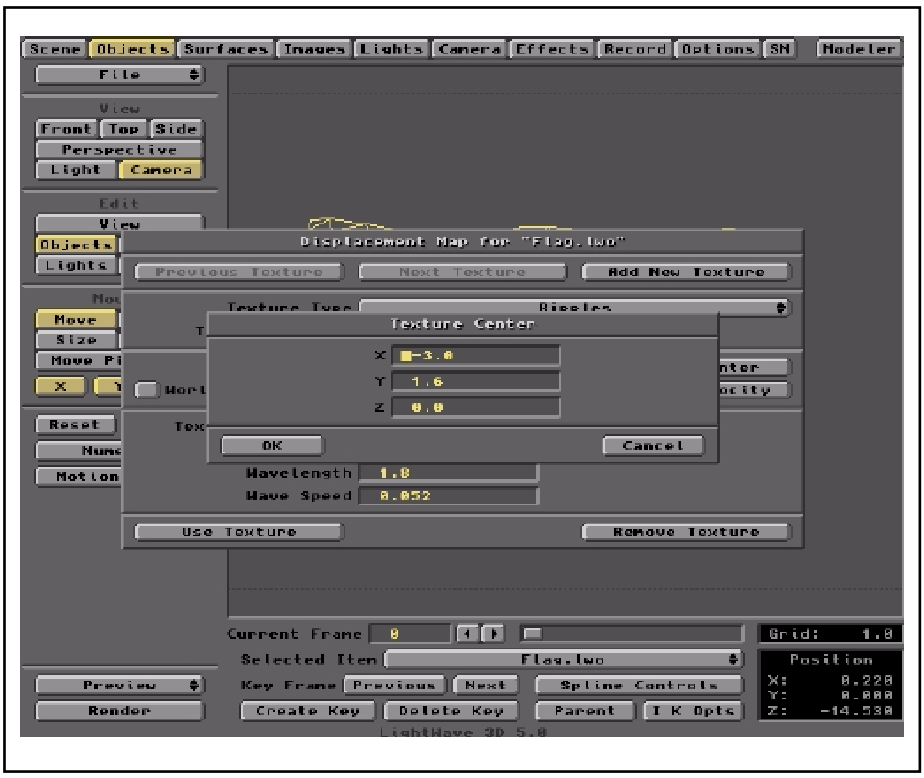
For some insight on projection axes, check out Tutorial 3.



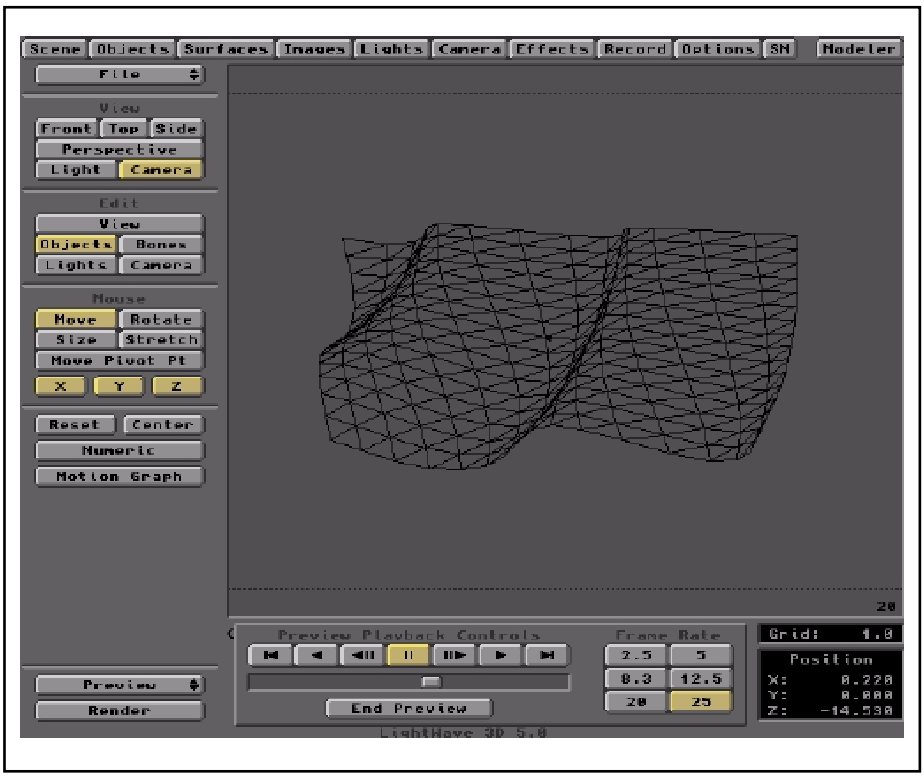
Other parameters to note are the Height Repeat and Width Repeat buttons. Deactivate these so you won't get any wallpapering of the image. The image must fit perfectly over the flag surface in 'one go'. Ensure this by clicking the Automatic Sizing button. LightWave will then stretch any image to fit the Surface if necessary. Making the flag the same size ratio as the image will avoid this, of course.

The next thing to do (if you haven't done it already) is set up the Ripples texture for the Flag.lwo. Go back into the Objects panel and click the Displacent Map button as described earlier. The pop up panel allows you to set the various Ripple parameters we've already considered.

The following screen grab shows the Texture Center pop up panel. I've placed the ripple Center to the upper left of the Layout stage. This will cause the waves to progress across the flag from top left to bottom right. You can ring the changes on this to suit your application.



When you exit the Displacement Mapping panel, you'll see that the wireframe of the flag.lwo is already textured by the Ripples routine. If you run a Preview animation render, you'll see how the wave action actually works. Here's what you should see if your Ripple parameters are about right.



Try a test animation by clicking the Preview button. A wireframe render will be created according to the parameters you use in the Render Preview panel. The anim controls will then pop up as shown above. When you run the animation, you'll probably see a glitch as the loop recycles back to the start. The easiest solution to this is to change the Wave Speed. Subtle changes to the Wave Speed setting will let you see the direction to go and you'll quickly get things looking OK. When you're happy with the Preview, it's time to sit back and let

LightWave do it's stuff. Note that the default Scene Editor will have already set things up for a thirty frame anim. Adjust this if necessary for the job in hand.

OK, now go to the Record Editor and set up the image format/anim type and the path for storing your rendered animation or frames. Check the Camera menu to see everything there is to your liking and click the Render button, go for a cup of tea, or take the dog for a walk, depending on the power of your machine. At some stage thereafter, you should get output something like this:

This image shows the black background LightWave assigns to all scenes by default This can be easily changed to a different colour or indeed to a background image, by going into the Effects menu. There you'll find options for changing the background in various ways. If you wish to use an image as a backdrop, first load it into the Images panel as described earlier. You can also add effects like Fog. These should all be tested out by experimenting with the settings, they're pretty logical.

For use on a webpage, you'll have to convert the Flag animation frames into Gif format. There are various freeware programes to do this (e.g. WhirlGif). Alternatively, pop over to the WaveGuide site's Downloads Page, where you'll find umpteen examples ready to use.

Now, what's the weather like outside? Is the wind blowing?

Tutorial 15: Organic Modeling with metaNURBS (LightWave 5)

How to create a Basking Shark from next to nothing

The easy way to create 'organic' models in LightWave is by metaNURBS modeling. Basic spline modeling is introduced in Tutorial XX , so if you've followed it you'll be familiar with a Spline Cage. The metaNURBS tool takes this to the next stage, by automating the process and incorporating many of the standard Modeler functions. The technique is more akin to modeling with a lump of clay, rather than gluing together small bits and pieces. Most of the commercial animations seen on film and tv are made by a NURBS type system. It's commonly called 'patch' or 'sub-patch' modeling. So, sharks or shellfish, shower-heads or shoes, they're no problem with NURBS (providing you have LightWave5).

Choppy Water

If you have ImageFX4 (one of many Amiga gems), you may have created the 'Choppy Water' anim from some live action frames hidden away in the Goodies directory. It's very nice, though you need a lot of RAM to do justice to the 600 jpegs supplied on the CD. When I'd put the anim file together using MainActor, I immediately thought it would make a great backdrop for a shark animation. So, the main problem was to construct a LightWave shark! I decided to make it a Basking Shark.

What's a Basking Shark?

Oh, come on! Have you never watched David Attenborough on the telly? Well a Basking Shark is pretty big. In fact, It's almost the biggest fish there is. Only the Whale Shark is bigger. Baskers typically weigh in at around five-thousand kilogrammes, give or take a few hundred kilos of plankton. That's what they eat. Indeed plankton is next to nothing, so they eat a lot of it. To collect enough plankton, they have some rubber-like extensible skin around the lower jaw. This inflates when swimming with the mouth open. It makes a colossal scoop with which the fish hoovers up the oceans. The plankton is filtered from the water through fibrous membranes on the inner openings of the gill slits. So there you have it. The Basking Shark....big, beautiful and utterly harmless.

Squalene

Squa...., what's this about? Well I couldn't talk about Basking Sharks without noting the way they've been exploited for hundreds of years. It's their squalene you see. Their livers are full of it and a basker's liver constitutes about a third of its body weight. Squalene is a rather exquisite natural oil. So, it was good for lubricating clocks, Spinning Jennies, sewing machines and such. For any chemists out there, squalene has the

empirical formula C30H50, determined by Tsujimoto in 1906. Its structure provides the generic description 2,6,10,15,19,23-hexamethyltetracosa-2,6,10,14,18,22-hexaene. Hmm..that's quite a mouthful, as Basking Sharks know very well. Intriguingly, Squalene exists in all animal cells, including ours and plays a part in virtually all life on Earth.

Squalene is a highly unsaturated hydrocarbon and burns very brightly. As a consequence, people working around costal areas have made a living by killing untold numbers of Basking Sharks for their livers. So, the oil wasn't only used as a lubricant. It was widely used in domestic and industrial lamps. The rest of the animal was chucked away to rot. Thank Goodness then, for Thomas Edison and his electric bulbs and Castrol, who showed castor oil was far easier to produce!

OK, let's make a Basking Shark!

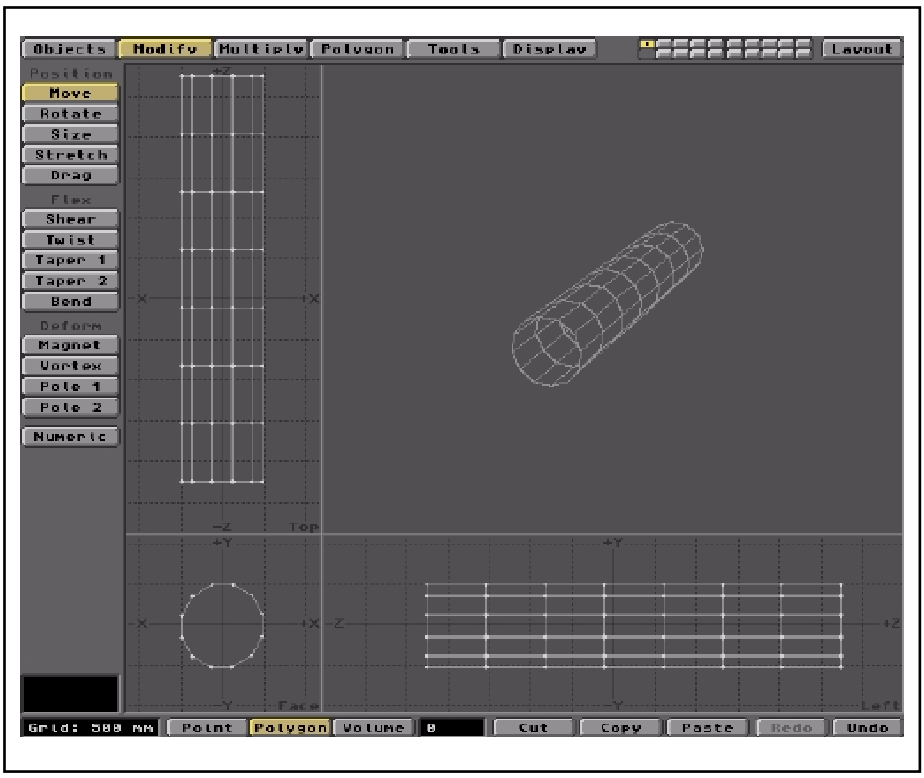
For Starters

For starters, we need a primary shape from which to mould the shark. The important thing to remember is we'll be employing the powerful metaNURBS tool, which only works with four-sided polygons. Well in LightWave5 it does. Later (pc/Mac) versions of NURBS work with three-sided polys as well. So, a primitive cube would do, but I reckon a cylinder is better. OK, using Modeler's Objects/Disc tool, create a cylinder by entering a Sides value of 12 and a Segments value of 7 in the Numeric panel. Orientate the thing as shown in the graphic below by selecting Z as the cylinder's axis. The scale can be anything you want, but to get it about right it should be a metre in diameter and say five metres long. So, set Bottom -2.5m, Top 2.5m, Radius X 0.5m, Y 0.5m and Z 5m. Or, just stretch out the bounding box to suite and click the Make button.

I chose these parameters for the cylinder after a fair bit of trial and error, using less and more Sides and Segments. I found it gives four-sided polys having the best size relative to the whole. They're easily manipulated into fins/tail of the correct size and their total number and orientation are also about right. By all means try different parameters, you can only learn by it.

Four-Sided Polygons

If you examine the cylinder, it's clear that the 'ends' have more than four sides (twelve in my case), so will not take part in the NURBS operation. That's not a problem, 'cos we're getting rid of 'em. In Polygon selection mode, select the two 12-sided ends, but make sure you de-select any adjacent quad-polys which got included in the operation. Now hit the Cut button to remove the ends. What's left is an open-ended cylinder made from twelve longitudinal rows of quad-polys. Good! It should now look something like this (note I've made the surfaces double-sided to improve visualisation).



Selected polygons on the surface of this cylinder will eventually become fins, tail, etc. A few of the 'head' points will also be manipulated to develop the shark's enormous mouth.

Heads and Tails

The cylinder should first be orientated so that two opposing rows (12 O'clock & 6 O'clock) running end to end are 'horizontal' and two rows (3 & 9) are 'vertical' (see above). This is easily done by slightly rotating the cylinder using the 'Face' view. Re-centralise the object's axis after the rotation. We need one or two polygons in the rows at the 'top' and 'bottom' of the cross-section to extrude vertically away from the body to create a dorsal fin and a tail. Similarly, we need the rows at each 'side' to be vertical, so two may be extruded out horizontally to make fins. I set the cylinder parameters so that there are also rows of polys at an appropriate angle (4 & 8) to extrude into those smaller lateral fins. These go between the main fins and the tail, but slope 'downwards'. Easy when ya know how, 'eh? Well, it took me around two hours to get this far, so don't be disheartened. You'll create a superb Basker!

Extruding Protrusions

I've talked about extruding polys to make fins, etc. What this involves is selecting a particular quad-polygon on the surface of the cylinder and drawing it out, to create a rectangular protrusion. This gives you a starting point for the metaNURBS modeling. It's certainly an extrusion process, but the operation actually uses the Bevel tool. The Extrude tool would be difficult for anything other than 'horizontal' and 'vertical' displacements 'cos it only displaces along the X, Y or Z axis. On the other hand, Smooth Shift (SmShift) allows you to extrude along the polygon's Normal, no matter what angle it's orientated. However, SmShift extrudes radially, so the extruded polys are likely to be bigger than their parent. Bevel retains the overall size of the parent polygon as it extrudes (Move) along the normal. However, it has the added function, Inset. The Inset value creates the chamfer on the end of the extrusion, which we don't need. So, why use Bevel? Well, it isn't only used to make bevels. In fact it's a very useful extrusion tool. How come? Well, if you make the 'Inset' value Zero, the 'Move' simply results in an extrusion, right? But before we start extruding, let's just think about fins and things.

Fins n' Things

Look at the cylinder and imagine it's gonna form the head and body of the shark. I decided to make the head on the left end of the cylinder as seen in the Left View. Assess where the main side fins will emerge along the cylinder. Knowing nothing of shark physiology, I reckon they're attached more or less on the centre line running head to tail, about a third of the way from the head. The dorsal fin emerges a little way behind the main fins, on the 'backbone'. The tail fins are easy, at the opposite end to the head, top and bottom. That's why we orientated the cylinder early on, so these extrusions are 'horizontal' and 'vertical'. That way they're much easier to work on. Each extrusion will need a single quad-poly, so mentally assign the ones to use.

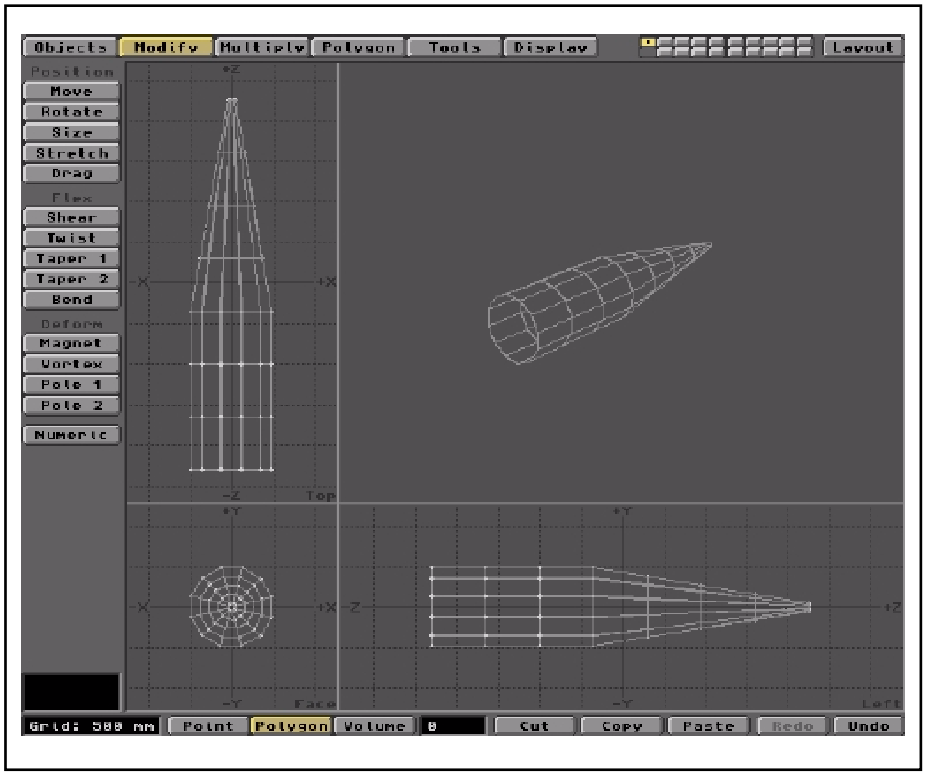
Now consider the small lateral fins which emerge about half way between the main ones and the tail. These are attached to the body at a point below the centre line. The cylinder has nice quad-polys in all the right places to make these fins n' things. If you need to make the model anatomically accurate, refer to a shark textbook before creating the cylinder. There may well be bits n' bobs I've left out because my shark will only be seen as a phantom, passing silently beneath the waves.....oooh it's gonna be so scarey!

Save it Now!

In case things go haywire when you start working on the body, Save the cylinder under a suitable name, say 'shark1.lwo'. Get into the saving habit. It will avoid a lot of grief and frustration! Better to save umpteen preliminary models than create a work of art and lose it accidentally - it happens so easily.

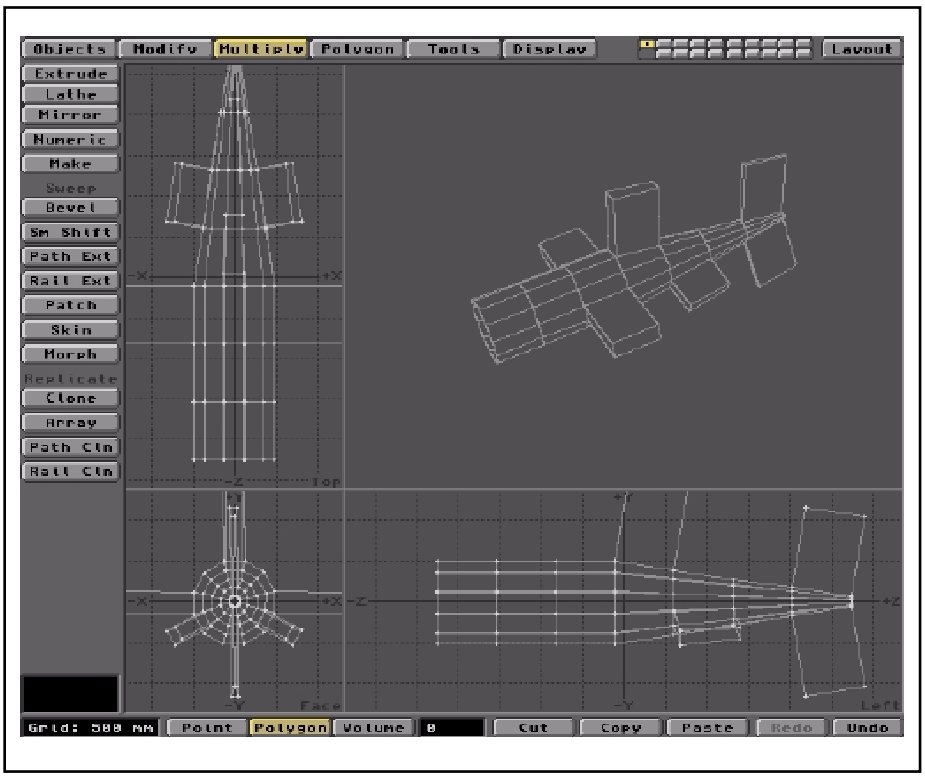
A Bit of Shape

First, let's give the cylinder more of a shark body shape. Let's taper it from about half way along to the tail. This is easily done using the Taper1 tool, but you have to be a little careful. First, select all the polygons in the tail half of the cylinder. Now go to the Face view window and place the cursor on the centre of the circle (it should be the axis origin). With the LMB, drag the cursor to the left. You should see the tapering take place. The far end will begin to close, but don't close it completely, leave a small circle. Note that if the original cylinder is not centered on the origin, the Taper will become asymmetric. Use the Undo button if things go wrong. Here's one I made earlier.



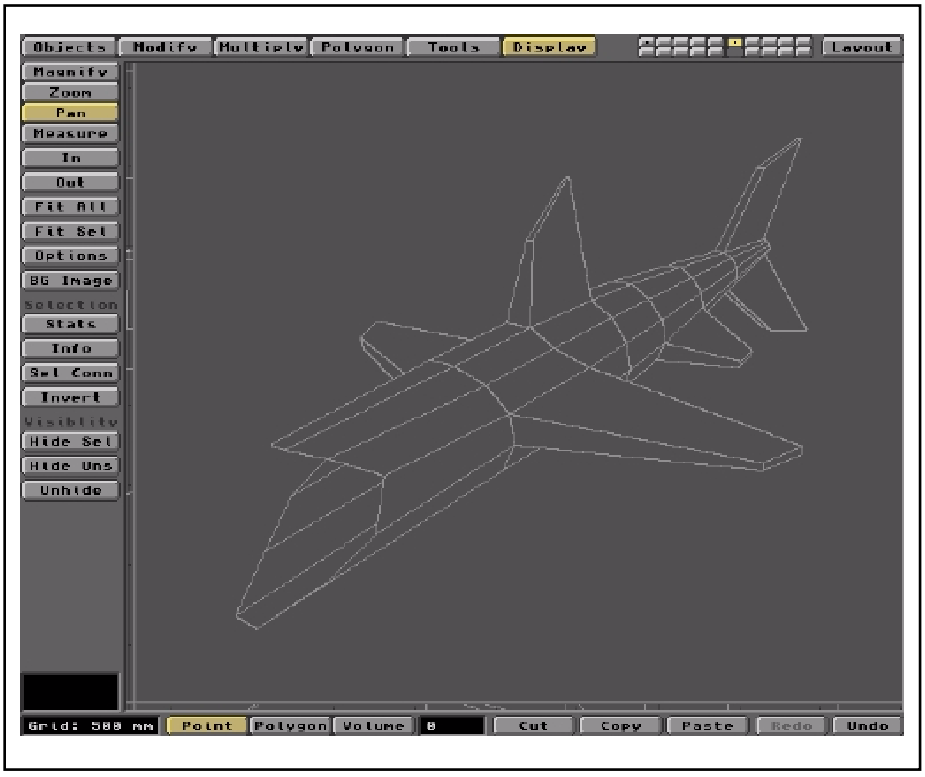
Let's get Beveling

In Polygon selection mode, select one of the quad-polys you want to extrude away from the surface. You can do them all individually to make them different lengths, or all at the same time for equal lengths. Just click the Bevel button under the Multiply menu. This pops up the Bevel parameters panel. The Inset should be set to zero to avoid the chamfer. For Move, type in a suitable value, say 1m for the big fins and the dorsal and a smaller value for the laterals but it needn't be too precise. Forget all the other settings. The result should look something like this...



Mouthing Off

It will help the development of the shark if you create a basic outline for its enormous mouth. This is easily done in Points mode. Select the points around the 'head' end of the cylinder and use Move or Drag to adjust the points as seen in the Left view window. Try to get a reasonable V-shape, but it doesn't have to be perfect. Don't superimpose any points in an attempt to get a sharp V-shape and certainly don't weld or reduce them in any way. Also, move some of the points at the outer ends of the fins and tail to get a more tapered effect. Remember, all the polys should remain four-sided. Aim for something like this...

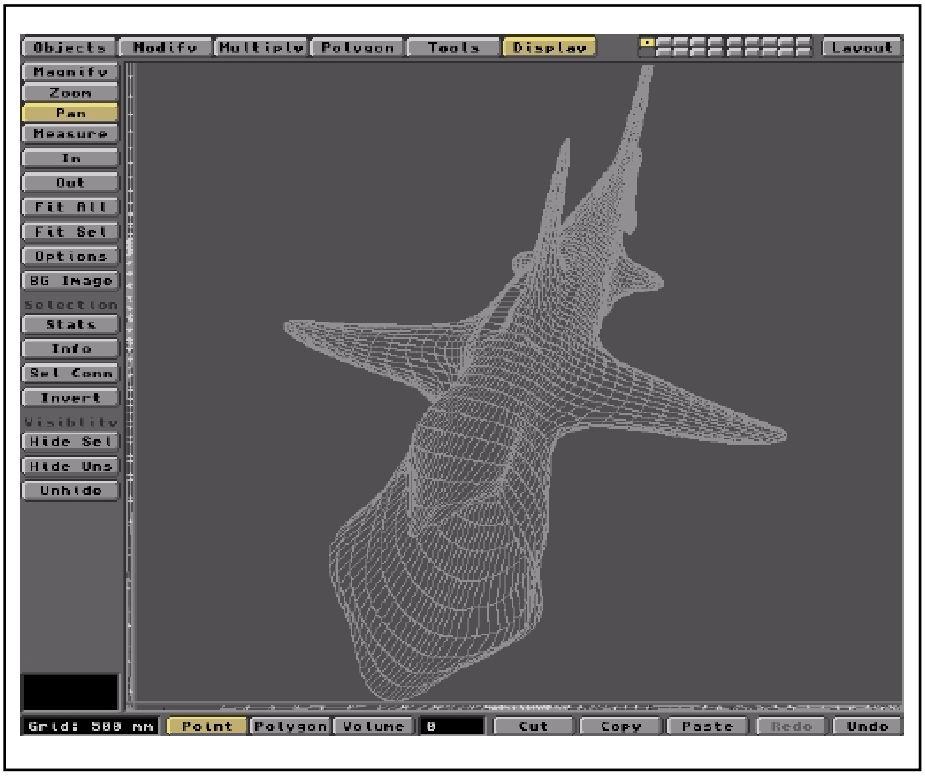


Non-Uniform Rational B-Splines

That's what metaNURBS are, but what they do is much more interesting. When you activate metaNURBS, each polygon is changed into a Spline Cage. These cages allow you to manipulate them using most of the tools you normally employ with polygons and points. However, they automatically assume optimal curvature between cages as they are pushed and pulled around. And you can make quite extreme changes without problem. This results in very organic and convincing models. MetaNURBS are activated by pressing the Tab key. That's the one with two arrows to the left of 'Q'. Once active, you'll notice an immediate change in the general form of the object. It's already getting the NURBS treatment. In some instances, this is all the treatment your crude model will require. Here, however, we must do a bit more work.

If you use a polygon modifying tool like Magnet, you'll quickly appreciate the special character of the NURBS object. Furthermore, they can even be saved in NURBS form for use at a later date. However, they have to be converted back into LightWave's standard polygonal format if you wish to use them in Layout. This is done using the Freeze tool, which will be found under the Polygon menu. After Freeze is applied, the spline cages revert back to polygons, the number of which is determined by the Patch Sub-Division setting. You'll find that in the Objects/Options panel.

A Sub-Division setting of 2 or greater will increase the number of quad-polys in the final object by an equivalent factor. This is something to be aware of if you have limited RAM or you wish to render the object in double-quick time. A frozen NURBS object will respond to the activation process as many times as required, though the multiplication of polygons should be bourne in mind. If you anticipate freezing several times, ensure the Objects/Options/Patch Sub-Division se tting is 1. Here's the frozen model after a bit of cage tugging using Magnet. Amazing eh? Remember to name the surface of this model something like SharkSkin and save it out.



Sheer Magnetism

I find Magnet to be great for moulding with metaNURBS. I can't go into too much detail here, but you must remember that Magnet is three-dimensional and its effects are determined by the size of the 'bounding box' you draw out when setting up the tool. Also, the position of the tool's cross hairs tells you where its effect is maximised. Always use the three view windows t ensure that the effect is applied where you want it and that the extent of its effect is correct.

Just a Shell

So far so good, but it's just a shell at this stage. One important area is the mouth, which shows the 'inside' of the animal. To look convincing, we need the inside area to render with a different surface from the outer skin. This means a complete 'duplicate' of the mouth area must be fitted inside the original, so we can assign a different

surface. I found the easiest way to do this was to select and copy to a separate layer, all the polygons making up the shark's head.

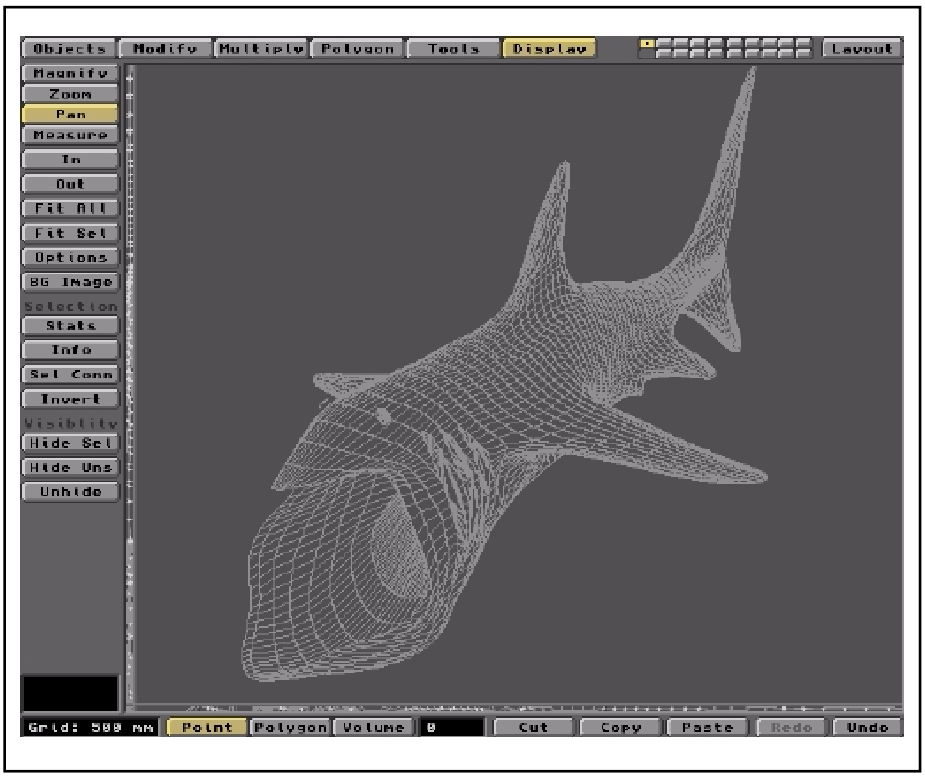
So, with the copy in the front layer and the original behind it, use the Size tool to reduce the size very slightly, say to 97%. The orientation may have to be adjusted so it will fit snugly inside the shark's head. Oh, don't forget to Flip all the polys in the inner part so they render 'face on'. Alternatively, make sure all the shark's surfaces have the 'Double-sided' switch 'on'. OK name the surface of the duplicate something suitable like SharkInside. Now copy or cut this layer and paste it onto the lower layer. Again, save the mesh under a new revision number. If you think your renders will see into the back of the mouth, you should close it up by pulling points backwards until they form a 'throat'. Weld points as appropriate. This will render a little rough compared with the skin, but for my purposes it was near enough.

The same thing goes for the 'lips' around the mouth. Close up, the model will show two separate surfaces with a space between them. If this is a problem, usie the Bevel tool on the mouth edge polygons to create a 'return' on the edge and provided a convincing 'seal'. It also gives some 'thickness' to the 'lips'.

A further refinement is the creation of gill slits and eyes. To make the gills, I used the stencilling technique described in the Tutorial ‘How to do Decals’ on the WaveGuide website. I created a suitably-shaped 'drill' polygon, extruded it wider than the shark, then stencilled in several gill areas. These were given a separate surface name (‘gills’!) and coloured black, so they appear like holes in the skin. My model has only three gill slits, though the genuine article has more (six I think).

Eyes were created from a couple of flattened hemispheres and given a black surface. And that was more or less that. The outer skin was given a blue-green mottled effect using a Fractal Noise texture. The inside of the mouth was assigned a 'marble' type texture in an attempt to emphasise the curvature.

Here's what the completed mesh looks like.



Now Tidy Up

Before you leave Modeler, you could tidy up the small 'hole' in the tail end of the body by joining all twelve points using the Weld tool. After this, just drag the resulting single point into the appropriate position. You may also find minor polygon errors created during the manipulations. These are usually obvious in a trial render. Alternatively, use the Display/Stats tool in Polygon mode to examine the types of polys you have ended up with. Most errors are easily resolved using the tools available under the Polygon menu.

Download it

If you can't be bothered creating your own Basking Shark mesh, you can download mine (155K) by going to the WaveGuide site's Download page. The mesh may need a minor adjustment to the tail fins (I noticed some small 'holes' in the render). The surface textures I have used are included in the archive. When this is unpacked, open the 'Surfaces' directory and copy the two files it contains into your 3D/Surfaces or NewTek/Surfaces directory, according to the version of Lightwave you are running.

Animating the Shark

In my website 'Baskin' animation, I used a duplicate shark as a Morph target for the first. The morph version has its main fins in a downward attitude. I created this from the first model using Magnet. By enveloping only the selected fin with the Magnet's box, you can gently draw the whole appendage downwards. It's important to create the morph target from the primary object so they have the same number of points. Anything else and LightWave will complain when you attempt the render. The animation setup has the two sharks morphing back and forth throughout the 320 frames.

You could achieve this 'swimming' motion using Bones. However, I decided on a morph because I wanted to use Bones for the body and tail movements. I felt so many Bones would be just too complicated to choreograph.

Anyway, here's a simple render of the Basker against a black background. Put him under water and he'll readily come to life, as second still and the downloadable animation confirms.

Cetorhinus maximus

Below is a typical composition . Placed in context using the 'Choppy Water' images, I used some Alpha channel compositing techniques to make the monster part in and part out of the water.

Frame 180 from the LightWave/MainActor anim 'Baskin'.

See the WaveGuide website for any further Tutorials not included in the manual.

Addendum

Modeler Functions which have no On-screen Button

Modeler has several functions which have no on-screen buttons to engage their operation. Some of these functions are associated with pop ups.

***Esc* (*Escape*)**

This is the keyboard equivalent of *Cancel*. It will close a dialogue box and return you to the previous environment providing you have not initiated dialogue by using the cursor. If a text field holding the cursor is involved, it will not return you to the previous editor, etc. until you disable the cursor by clicking outside the field.

***Enter* (*Enter*)**

This is the keyboard equivalent of the *OK* button. If a text field is involved, it will not return you to the previous editor, etc. but will move you to the next field if there is one. If there is no subsequent field, the pop-up will be closed. When a menu or a requester is involved, *Enter* has the same function as the *Return* key.

***Ctrl* (*Control*)**

The *Ctrl* key is used as a mouse movement modifier. If you hold down *Ctrl* while dragging the mouse, its action will be constrained to a specific axis (eg. *Move* , *Drag*), or it will move in specific increments (eg. 15° of rotation with *Rotate*, *Twist*).

***Ctrl + a* (*Fit Seleted Item to Window*)**
This key combination will cause the selected item only to be resized to fit the *Current Window*.

***g* (*Go to Centre*)**

This key press will centre the area of the screen currently beneath the cursor. Can be used as a version of *Pan*.

***j* (*Jump*)**

The *j* key causes the currently selected *Point*(*s*) to 'jump' to the coordinates of the cursor.

***r* (*Rotate Right by 90º*)**

All selected items will rotate 90º to the right (i.e. clockwise) relative to the position of the cursor.

***e* (*Rotate Left by 90º*)**

All selected items will rotate 90º to the left (i.e. anti-clockwise) relative to the position of the cursor.

Keyboard Equivalents of Modeler Buttons

Remember that pressing the *Help* key at any time will pop up a list of keyboard equivalents for the buttons found on the current *Editor.*

Known Bugs

There are several bugs in the LightWave software, some of which were fixed by a patch incorporated into version 3.5. Others remain and the following may explain any erratic behaviour you experience.

For example, in Layout, load a picture you wish to use as a *Background Image* in Modeler. Open Modeler, then select the *Display* menu and choose *BG Image*. Select *Z Axis* and then select the loaded *Image* for use. The *Image* will appear in the bottom *Edit Window*. If you now *Zoom* in a few times (using the *In* button, or the **>** key), the screen will corrupt and the programme will eventually crash.

This and other problems can be cured by running the programme in NTSC mode. Do this using the Amiga's *Early Startup* control system. Hold down both mouse buttons while booting. This brings up the *Early Startup Control* panel, in which you can select the video mode you wish to use. Select *NTSC* under *Display Options* and then let the machine continue with the load Workbench. If Workbench boots up OK, run LightWave and reload the *Image* as before. Enter Modeler and you will see a strange wipe effect not observed in PAL. Display the *Image* in the *Background* and the *Zoom* will now operate properly.

If you use *Mode Promotion* to enable *Double PAL* mode to be used, then the *Moving Preview* option under Layout's *Display* menu will cause sceen break up. This cannot be fixed in *Double PAL* mode. See *Screen Mode* (page 3) for information on *Mode Promotion*.

The *Patch* command (page 86 and Tutorial 9) requires that a *Merge* or *Weld Points* operation is conducted on the overlapping *Curves*. This step seems to have a bug, in that you may get a persistant message that the *Points* must overlap, even when they do. This problem was fixed by a 'patch' in version 4.0, which also seems to fix the problem in version 3.5. However, Tutorial 9 was created as written, with no v4 patch and without the problematic message. If this bug presents you with difficulties, please contact me for a copy of the v4 patch.