Art attack

Benjamin Woolley contrasts photorealism techniques in rendering 3D graphics, and previews Riven, a new game by the Miller brothers, which ventures into the realms of art.

I attended a lecture given by a 3D graphics pioneer at the SIGGRAPH graphics conference a few years ago, on a new form of rendering called "radiosity". He showed an image displayed on a PC sitting on a desk in an ordinary office. Not only the image on the screen was synthetic, he told his audience with pride; so was the screen itself, the desk, the office and everything. To me, it was obvious that the scene was synthetic: the human eye has the ability to spot even the most subtle clues that give the game away; it is something about the light, the composition or the perspective. The image may look photorealistic but it does not look *real*.

Radiosity is very much an engineering solution to the problem of "photorealism". In fact, it has its origins in thermal engineering and works on the principle of calculating the transfer of radiation between surfaces. In graphics terms, this means calculating the way light radiation bounces off one face and onto another. The result is "photorealistic" because radiosity can more accurately reproduce the diffusion of light present in physical environments: the phenomenon of "colour bleeding", for example, where the colour of one object bleeds over to those that surround it (picking up its reflected light).

Radiosity is beginning to reach the mainstream 3D market. Lightwork Design <www.lightwork.com> has a radiosity renderer which Kinetix will be shipping as a plug-in for 3D Studio MAX.

Intelligent fish

The fetish for photorealism represents one view of 3D graphics but an alternative, and more interesting, approach goes in the opposite direction. The surrealistically-named company, ThinkFish, was founded by MIT Media Lab's Rolf Rando with the aim of introducing a little more art to the science of 3D graphics. The means of doing this is a technology dubbed "LiveStyles", a rendering system which goes in the opposite direction to radiosity. ThinkFish calls it a Non-Photorealistic Renderer (NPR); which is, the company claims, "intelligent".

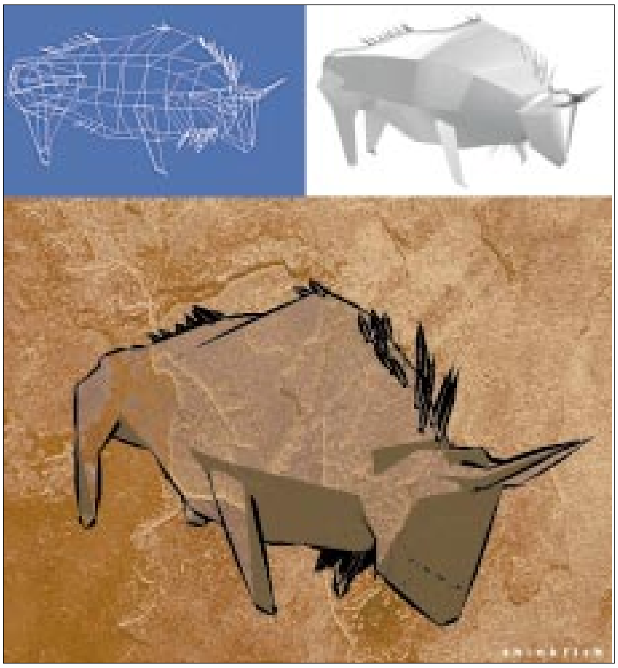


Fig 1 The ThinkFish "intelligent" renderer at work, picking out the key lines from a simple mesh

It examines a model to establish its "key lines" (Fig 1) and uses these to create the rendered image. It does this in the style of a cartoon, a pencil sketch or a paintbrush, or in any of an infinite number of different styles (the LiveStyles) which the user can select to produce the desired look.

The result is more like a 2D drawing than a 3D scene. There is no pretence at realism nor any attempt to create a feeling of depth or space. Rather, the user is encouraged to use their imagination to come up with something unique, something with the advantage of being a 3D scene which you can explore or animate.

ThinkFish's commercial strategy is to licence the technology to different graphics applications developers to include in their products and sell packages of different LiveStyles (Fig 2), "from Picasso to the Simpsons", to end-users for $30-plus.

At the time of writing, it had gained the support of Apple, which has announced a LiveStyles plug-in for applications which use QuickDraw3D (check with the excellent QuickDraw3D website at quickdraw3D.apple.com for news of availability). Fractal Design (to be renamed MetaCreations if its merger with Metatools goes through) also intends to include LiveStyles in the next release of its 3D products including Ray Dream Studio and Poser. A company called Vertigo has shipped a selection of LiveStyles with its Adobe Photoshop plug-in, Dizzy 3D (at the time of writing, only available for the PowerMac).

One of the benefits promised by the LiveStyles technology is that it will sidestep one of the great problems with 3D graphics: the hunger for hardware resources. The more photorealistic you try to be, the more

p278 ➢

Fig 2 (left) Examples of the same scene in two different “LiveStyles”

Fig 3 (below) A preview of Cyan’s Riven, taking to a new level the beautiful detailing that was a hallmark of its predecessor, Myst

ravenous this hunger becomes. Radiosity, for instance, depends not just on calculating how much light each shape will pick up from the light sources in the scene but how much of that light will be radiated back into the scene and picked up by other shapes. LiveStyles, in contrast, works by simplifying the scene and drawing it in the broadest brush strokes. As a result, Thinkfish claims that LiveStyles will allow fast, smooth, real-time rendering of the most complex scenes on standard PC hardware.

The Millers’ tale

One scene that could never be rendered in real time on even the most powerful mega-specified supergizmo workstation is the island of Myst, the setting for the now legendary game from those grand wizards of game design, Rand and Robyn Miller.

I remember my first encounter with Myst: that luminous sky, those craggy rocks with huge iron cogs erupting from them, those elegant buildings and soaring conifers and the sound of water lapping and seagulls calling. Over three million copies of Myst have been sold since its launch in 1993 and the game has created a look that no other has equalled, except for Riven (the sequel to Myst). Thanks to CNET’s Gamecenter <www.gamecenter.com> and other sources, like the unofficial Riven site <members.aol.com/mystsequel/index.html>, we have peeked at Riven, and the graphics look even better.

The secret of Myst’s success, in my view, was the Miller brothers’ decision not to go for the real-time rendering you get in Doom-style arcade games. Real-time rendering means the player has total freedom to explore and interact with the world the game inhabits. The downside is that the whole look of the thing has to be sufficiently simple to be created, frame by frame, 25 or more times per second on standard PC hardware, which results in the sort of gaudy graphics you get in Quake.

The Millers took a different approach. They saw rendering as part of the production process — part of their art, as it were. This means each possible view of the world being explored has to be pre-rendered, which limits flexibility. It also means each of those views can be exquisitely detailed and carefully composed. A typical, and for me, beautiful, example can be seen in Fig 3.

Myst was very much a Mac-based product, created mainly using Stratavision 3D. Riven was generated using SoftImage running on Silicon Graphics Indigo workstations — in other words, Hollywood-grade hardware. Also, according to CNET’s report on Riven, Cyan (the Miller brothers’ company) has produced its own shaders to extend the level of surface detail achieved in Myst. Despite having access to all that extra grunt, the best thing about Riven looks like being the revival of Myst’s vision, the sort of 3D graphics that go beyond realism and into the realms of (dare I suggest it) art.

Models get personal

Occasionally I receive plaintive requests from readers wanting to know where they can get hold of clip 3D models for their projects. The good news is that there are huge libraries of such models available on the net. The bad news is that you have to pay for many of them. Thankfully, Avalon <avalon.viewpoint.com> is still free, and has a search engine which you can personalise to your own requirements. Avalon has a library of objects in most file formats and a selection of handy utilities. There are categories ranging from aircraft to dinosaurs, and models ranging from apes to the Venus de Milo, all free, but of varying quality.

If you are prepared to pay for your clip models, you will find a wider variety available. Viewpoint (which manages Avalon) has an enormous variety that can be browsed online (though not, when last I looked, purchased). Two other sites I managed to find are the REM 3D Bank <infografica.com/3dbank/s2.html> and the Marketplace <www.3dsite.com/marketplace/>. The latter allows online ordering via a secure website.

Benchmarks

If you are thinking of buying a 3D graphics accelerator card, have a look at Fourth Wave’s benchmark site at www.fourthwave.com/3d-perf. It provides technical details on different benchmarks for measuring 3D performance, and a list of test results.

PCW Contact

Benjamin Woolley, writer and broadcaster, can be contacted at **3d@pcw.co.uk**.

AGP angst

Advanced Graphics Port — you can either take it seriously as an important graphics standard of the future, or you can ignore it. Benjamin Woolley wonders which is best.

Sometimes it is hard not to believe that, when it comes to marketing, the PC industry is about as principled as an Albanian pyramid seller. How else can you explain the announcement of MMX just after Christmas, or motherboards being shipped without USB support?

If you are interested in 3D graphics, there is a strong possibility you may be caught in a similar trap, and the reason is another little triplet of letters courtesy of Intel: AGP stands for Advanced Graphics Port, and we are now beginning to see a host of new graphics products being announced bearing those initials.

For instance, leading graphics hardware companies like 3DLabs, ATI and S3 have announced "AGP-compliant" chips. The first will be offering the Glint Gamma processor, ATI the 3D RAGE PRO and S3 the Virge/MX and GX2.

So what *is* AGP? And, when it arrives later this year, is it going to render non-AGP PCs as obsolete as non-MMX ones will no doubt prove to be? To address the latter question first, the obsolescence risk factor could be higher since AGP, like PCI, is effectively part of the PC's architecture. If your PC motherboard does not support it, you will be scuppered.

AGP was announced by Intel a year ago and marks an important step in the development of the PC as a 3D graphics platform. It effectively (although not literally) creates a Unified Memory Architecture (UMA). When Silicon Graphics launched its O2 workstation, UMA was one of its key features and one of the reasons it was so cheap (by Unix workstation standards).

In conventional systems, 3D graphics boards have to use limited on-board specialist graphics RAM to store the depth, colour and transparency information (the z-buffer, texture buffer and alpha-buffer, in technical terms) that goes to making up a 3D scene. Such RAM is expensive and there is never enough of it.

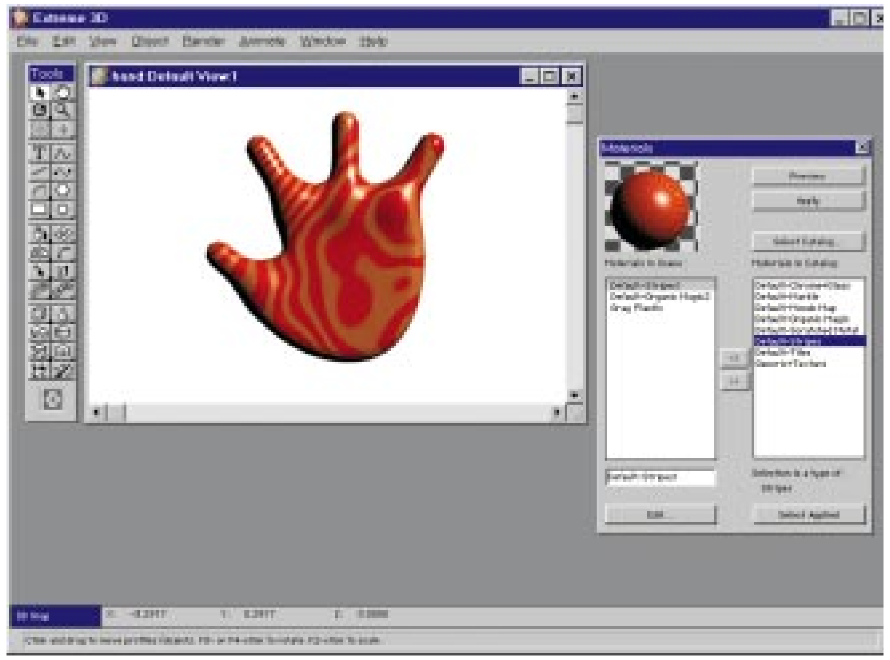


Fig 1 Extreme 3D's metaform used to create a balloon hand

AGP aims to solve this problem. It is a fast bus offering the 3D graphics accelerator chip direct access to the host system's RAM. 3D graphics applications can thus use any or all available memory as a unified block to build up each scene, enabling much larger scenes and much quicker rendering.

There seems little doubt that AGP will become an important graphics standard as 3D becomes increasingly pervasive on mainstream PCs. In a few years' time it may even be essential to run the latest 3D applications. Unfortunately, it is almost impossible to tell how many years. Like MMX, USB and like Windows 9 (?), the manufacturers have given no clear launch dates nor cost information.

Intel originally suggested it would begin to make its mark early this year but currently there is still no sign of AGP motherboards or graphics controllers. Most AGP chip vendors now talk in terms of shipments beginning in the second half of 1997. It could take a lot longer. It could suddenly pop up after Christmas, once many people have invested in non-AGP systems. So what is an upgrader to do?

Then again…

One possibility is to ignore all future developments and take advantage of the plummeting cost of systems that rely on older technology. It is generally agreed that the benchmark 3D platform at the moment is a 200MHz Pentium Pro with 64Mb of RAM, a graphics accelerator based on the 3DLabs Glint chip and a fast and wide SCSI interface driving a monster hard drive.

These systems are coming down in price, so now is a good time to get one. I spent a month using just such a system, Compaq's new Workstation 5000 installed with Windows NT 4.0 and I can report that it is perfect for modelling work, using applications like 3D Studio MAX.

However, if you don't have the budget for such a radical upgrade, don't despair. You can do a great deal simply by swapping your video card. I've been trying a Diamond Fire GL 1000 slotted into my old Compaq Deskpro. It was a nightmare to install, requiring a new BIOS for the board, a new set of display drivers and, eventually, a new operating system (I could not get all the Win95 drivers to work, so I had to swap over to NT 4.0). Once I had it up and running it meant that, even with an ancient Pentium, the system could display properly-shaded and lit models in a preview window that were rendered in more or less real time.

Boards like the Fire GL with 4Mb of video RAM (£285 ex VAT) use accelerators, like the 3DLabs Permedia NT chipset, which are aimed at the intermediary graphics market. So it may be worth upgrading the graphics controller only, before you dump the whole system.

Extremely frustrating

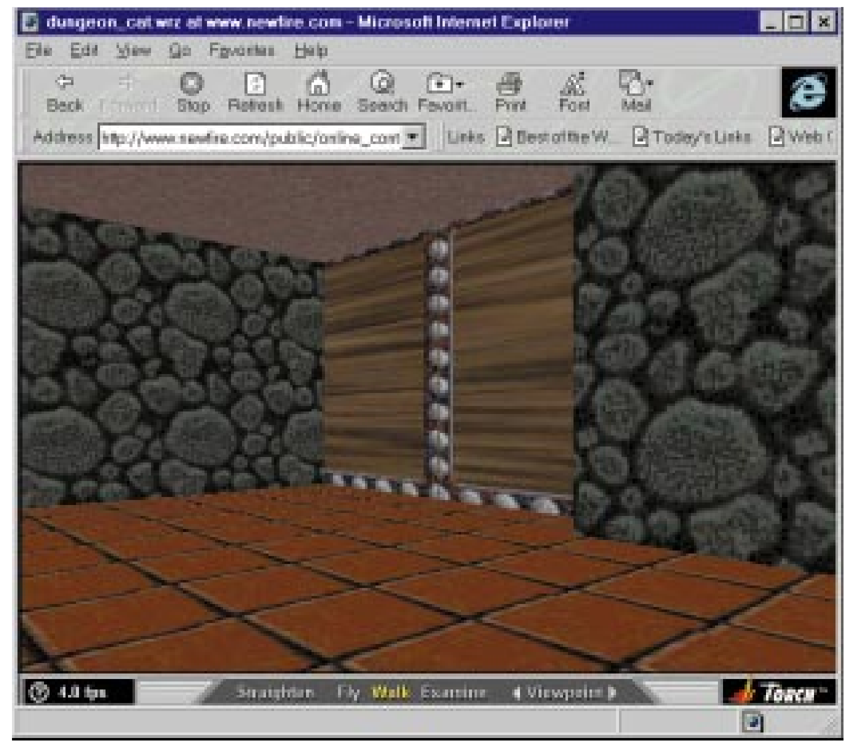
Extreme 3D from Macromedia is an important product. At around £500, it is priced substantially below full-blown professional packages and its specification now almost equals many of them. The recently-shipped version 2.0 now comes with a particle system and an implementation of a modelling method generally known as "metaballs".

Particle systems (see *PCW*, November 96) allow you to create clouds of particles (snowfalls, dust clouds, hailstorms) that are animated along a specified trajectory, with a specified degree of turbulence over time. The one supplied with Extreme 3D 2.0 is quite sophisticated and relatively easy to use. There is no library of presets, however, which is a pity because it takes a lot of time and practice to get the dynamics of a particle system to work (because you usually need to see the fully-rendered animation to establish whether the required effect has been achieved: low-res partially rendered previews are rarely adequate).

The metaballs modelling tool, called "Metaforms", was, for me, a more interesting addition, because 3D Studio MAX, which costs around five times more than Extreme 3D, does not include such a tool in its standard set. Metaforms (Fig 1) allows you to create simple shapes and fill them with a sort of virtual putty that you can then shape to create rounded, organic forms. Unfortunately, when it came to using the thing I suffered some sort of imagination crash; I could *not* think what to do with it. I tried dinosaurs, dolphins, cartoon characters and in every case ended up with something that looked like the sort of elongated balloons entertainers twist into ingenious shapes at kiddies' parties.

Extreme 3D has improved with this new release. It offers welcome and particularly strong support for VRML (including version 2.0). And the network rendering, which can be used to good effect even over small LANs (even those with a mix of PCs and Macs), is a boon. But, in my judgement, it still suffers from an awkward interface. For example, you can only swivel a view along one axis at once, which is very frustrating when you are trying to get a feel for an object's geometry: I find other budget packages, such as Truespace and Ray Dream, much easier in this respect.

Furthermore, Truespace version 3, which should be shipping by now, threatens to outclass the opposition with the promise of collision detection, its own implementation of metaballs ("Live Skin"), a way of moulding models called "Plastiform" and materials that give real physics (weight or elasticity) to the objects to which they are applied. It sounds exciting and likely to give Extreme 3D a run for its money.



The Heat is On

A number of companies are trying to find ways of pepping up VRML 2.0. Many, like Black Sun, have concentrated on the idea of developing it to help foster virtual communities. A new name, Newfire, is more interested in straightforward gaming. It has just announced Torch, a player which it claims can turn VRML worlds into Quake-style games (the example shown above is from a "Dungeon" demonstration game). Torch is designed to work with Direct3D, so it should make use of any on-board 3D acceleration with DirectX drivers. The company claims that as a result of this and a 3D engine that "carefully eliminates unseen polygons", Torch is four to eight times faster than other 3D internet players. Judge for yourself at www.newfire.com.

PCW Contact

Benjamin Woolley, writer and broadcaster, can be contacted at 3d@pcw.co.uk.

On top of the world

Benjamin Woolley and Bryce build a world in seven days: see how they set about their mountainous task. Ben's design style could be hampered, however, by the lack of file formats

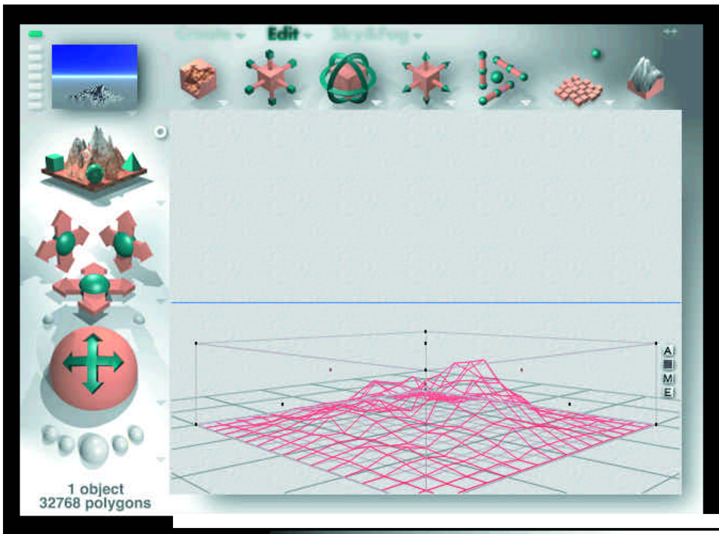


Fig 1 (left) Bryce 2's artistic interface

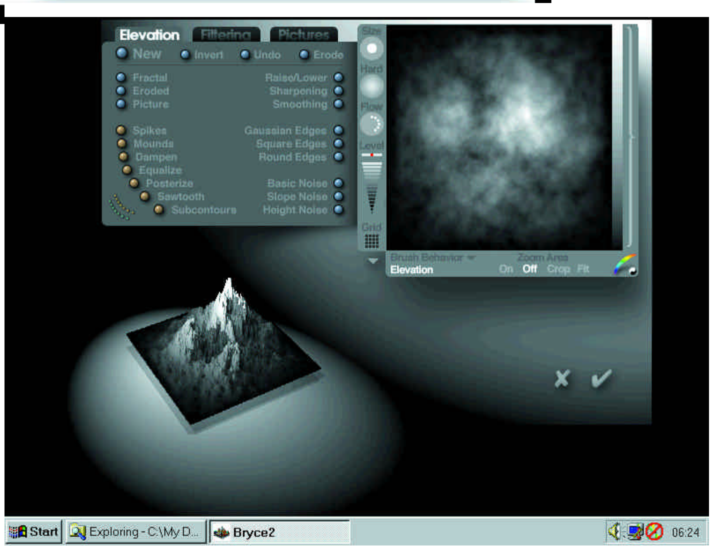


Fig 2 (below) Bryce 2's "terrain editor"

It was an ambitious project. For the last episode of BBC2's *The Net*, I decided to have a go at building the world in seven days using nothing but 3D Studio Release 3 running on my Compaq Deskpro XL (which had what then seemed like a warp-speed 66MHz Pentium and a vast 16Mb of RAM).

I built up the world around me: a desk, a room, a fireplace, and a window overlooking a forest and snow-capped mountains. For the finale of this spectacle, the (virtual) camera zoomed out of the window, up through the soaring trees, up through the plumes of magnificent fireworks, and then turned to peer down as we pulled away into space, watching the mountainous terrain recede until we could see only continents and, finally, a globe like our own Earth floating in the speckled firmament. All this was done to the sound of the incomparable Sachmo singing It's A Wonderful World.

It wasn't a wonderful experience. Day in, night out, I had to re-render each sequence, then re-render the re-renders. Nothing went right, nothing. That is, except the bit I expected to be most difficult: building my virtual world's mountainous terrain.

One of the plug-ins then just released for 3D Studio was called Displace. You started off with a flat plane split up (tessellated, like a mosaic) into a fine grid. Each intersection in the grid represented a vertex, a point in the geometry. Over the top of this plane you mapped a two-dimensional image. This image was created by a fractal generator, which produced what looked like a black-and-white satellite image of a mountain range: peaks of white fading away to valleys of black.

The displacement plug-in used this fractal image to calculate how much to displace — or elevate — each vertex in the flat plane. The vertices mapped to the bright pixels were elevated the most; the vertices mapped to the darkest ones were elevated the least. The result was surprisingly natural-looking geology, produced in an instant. By exporting the fractal image to a paint program and adding colour to it, I could also create an accurate texture map to drape over the newly generated range, knowing that the rocky screes, grassy plains and snowy peaks painted into the picture would settle exactly onto the correct bits of the geometry.

I did not have time to get the terrain as richly textured as I hoped, but it did make me appreciate 3D software's potential to produce breathtaking natural vistas without demanding breathtaking skills and resources. Enter Bryce from software house,

p278 ➢

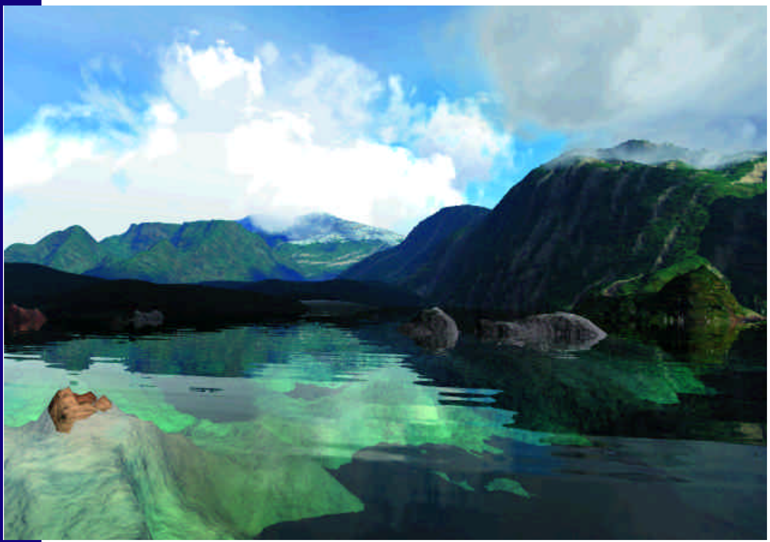


Fig 3 **One of the samples supplied with Bryce 2. It's called Scotland, and is the work of Kai Krause, Metatools' resident guru**

Metatools, the second version of which is one of the most enjoyable 3D tools around. Bryce falls into a new category of software product that is becoming increasingly common in the graphics market: plug-ins that have gone solo. Fractal (which, at the time of writing, was planning to merge with Metatools) did this with Detailer and Poser. Metatools did it with Goo and Bryce. Goo is for stretching and distorting bitmaps, and is firmly aimed at the recreation market. Bryce (named after a canyon in Utah) is for generating landscapes. It does it using the same basic principle as the 3D Studio Displace plug-in, but with some clever embellishments and the prettiest interface you've ever seen.

Firstly, the interface (Fig 1). It breaks all the conventions of the Mac (the platform for which most of Metatools' products were originally developed) and Windows. This isn't a dull desktop you're working on. Nor is it the sort of engineering studio-cum-nuclear-power station control room you get with products like 3D Studio MAX. It is, flatteringly for those of us who aspire to being artists, a studio. The icons pulse seductively when the pointer strokes them, giving a teasing hint as to what will happen if you touch them. The menu items are in soft focus and glow when you select them.

Behind the interface lie three main components. In conventional 3D parlance, you would call them a scene builder, a modeller and a materials editor. The scene builder allows you to create (from a wide selection of primitives) and manipulate objects — in particular, planes. A landscape is, when you think about it, a set of objects arranged between two infinite planes: a ground plane and a sky plane joined in the middle by the horizon. Each Bryce scene has these by default. You can also have infinite water and cloud planes that sit in between. You can then add a number of finite planes, perhaps one big one in the background that acts as a mountain range, and a smaller one in the foreground that represents the foothills. Where the peaks of the mountains poke through the cloud plane they are swathed in mists, and where the valleys dip beneath the water plane they become submerged beneath lakes.

To edit these planes you use the terrain editor (Fig 2), Bryce's main modelling tool. This is basically a bitmap editor for manipulating greyscale displacement maps. The window in the top right of the screen shows the map. To the left is a panel of tools for changing it, including ones that will add "erosion" (lots of little black cracks that creep in from the edges), raise or lower the elevation (increase or decrease the brightness), add noise, and so on. You can, of course, import bitmaps (created using a paint program like Photoshop) and even mix two together. The 3D black-and-white mountain range in the bottom left of the editor shows what the resulting terrain will look like, updated in real time. This sample terrain can be rotated using the mouse, so you can see it from all angles.

You texture these terrains using a type of material unique to Bryce 2. It is called a 3D texture, and the explanation in the manual is so paltry I didn't understand it. Suffice to write that the way the texture is applied to an object changes depending on the object's height and the angle of its sides. If the object is in the shape of a mountain, one texture can be used to put a white snowcap on its peak, a brown rock face on the slopes, and grassy cover on the plateaux.

The Materials Editor is not nearly as easy to use as Bryce's other components. For one thing, the terminology in the manual is non-standard. For another, trying to figure out how a 3D texture will be applied is about as intuitive as quantum mechanics. Thankfully, there is a generously stacked library of ready-made textures supplied with the CD, and, at extra cost, there is an Accessory Kit with more samples of both textures and terrains.

File formats

My dream is that products like Bryce will become the norm in the 3D world, replacing monstrous applications like 3D Studio MAX and Lightwave. Plug-in architecture is all very well, but it is expensive and cramps developers' design style. Instead, it would be much better to have separate applets: a selection of renderers, texturers, scene builders and modellers, each one with particular strengths for particular jobs.

There is one large obstacle standing in the way of this vision: file formats. Currently, there is no single standard for interchanging 3D data sets between graphics tools. This is partly because of proprietorial protectiveness of the software houses, but the problem goes deeper than that. With programs like Bryce having rendering novelties like 3D textures, it can be technically difficult to translate the resulting file into another format without losing important information. The most common interchange format in the PC world, DXF, is not up to the job, as it was developed centuries ago by Autodesk for CAD files and is really only suited to swapping untextured objects and meshes.

I do not know if it is possible to create a standard format that is capable of embracing all the novelties that products like Bryce 2 and Poser are bringing to the market. VRML 2, being extendible, may be up to the job. Apple's 3DMF, the format developed for its QuickDraw3D API is popular with companies like Fractal and Bryce (which have their roots in the Mac world), and it is flexible, so that may be one to consider.

Whatever happens, until a powerful interchange format emerges, the benefits of products like Bryce 2 will remain locked in their own little worlds.

PCW Contacts

Benjamin Woolley, writer and broadcaster, can be contacted at 3d@pcw.co.uk.

World in motion

It's a funny old world — or at least, it looks very different in 3D than the picture-book 2D views we're familiar with. Benjamin Woolley sets his sights on a more accurate projection.

The picture we have of the world is one that is fundamentally distorted because it is a two-dimensional version of a three-dimensional surface. If you look, for example, at the standard map of the world, the so-called "Mercator Projection", China appears to be roughly the same size as Greenland when in fact it is four times larger. This distortion occurs because the land nearer the poles is stretched out to the width of the equator (to form the rectangular shape of the map), so countries on the equator appear narrower than they should when compared to those closer to the poles. You can see how this happens in Figs 1 & 2. Fig 1 shows a map of the world. Note how huge Greenland is compared to China. Fig 2 shows the same map wrapped round a sphere, with Greenland now assuming its proper proportions. (I created the globe using Fractal Design's new Detailer package, of which more later.)

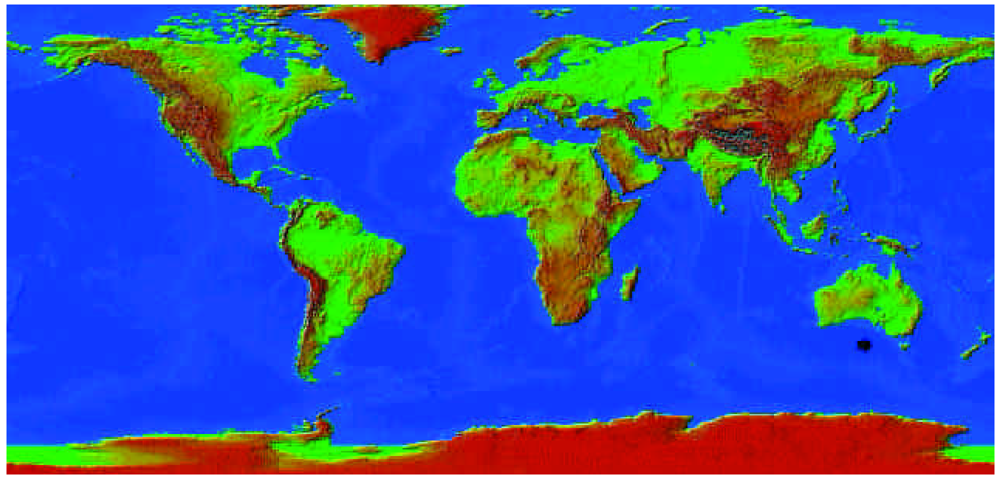


Fig 1 A texture map of the world. Note Greenland's size relative to China

There have been various attempts to produce more accurate projections (one of the best is said to be the Peters Projection, which makes Africa and other equatorial landmasses look huge, and more polar places, like our sceptred isle, teeny — you can have a look for yourself by browsing www.webcom.com/~bright/table.html), but none of them can be perfect. In the transition from 3D to 2D, something has to go, and in this case it is the true size and shape of each country.

As I have discovered from my email inbox, such problems are not confined to geography. A number of people have described the problems they have encountered trying get their texture maps to work, so I thought this month I would concentrate on this most perplexing area of 3D artistry, and at one tool that claims to make it easier.

Generally speaking, when you are trying to create a 3D scene, the sort of project you are dealing with is the reverse of Mercator's: you are trying to turn a 2D image into a 3D one, to take your flat map and wrap it round a sphere or, more usually, an irregular, complex shape. If you take another look at Fig 2, you can see quite clearly one of the first problems you encounter when trying to do this. Greenland's shoreline is slightly fuzzy, and there are two reasons for this. The first has to do with the size of the map: it has fewer pixels in it than there are on the surface of the object as seen from this perspective and at this size. You encounter this problem regularly, most obviously when the 2D bitmap, the texture, is placed on a wall or floor receding into the distance. As you can see in Fig 3, the bitmap is blurry at the point where the wall comes closest to the point of view. The solution to this problem is to match the texture's resolution to the wall's at the point closest to the camera. This means actually working out how many pixels there are down the edge of the wall, and making the appropriate edge of the bitmap the same number of pixels in size (in this case the bitmap is tiled, so I can divide the number of pixels in the rendered scene by the number of repetitions of the texture across the height of the wall).

The second reason for Greenland's blurriness is that where the map is approaching the poles, it is getting progressively scrunched up. There is no way of completely overcoming this problem unless you somehow manage to create a bitmap with progressively lower resolution towards the top and the bottom of the image. As far as I know, no image file format supports such variable resolution.

How, then, can you keep such distractions — "artefacts", as they are called in the business — to a minimum? By getting a grip on the way your 3D package projects or "maps" the texture onto the object. In all 3D packages there are basically three ways of mapping, usually known as spherical, planar and cylindrical. Spherical mapping is the sort demonstrated with the map of the world. Planar projects the texture onto the object as a film image is projected onto a screen. Cylindrical winds the image around an object like a label round a tin of beans. You can generally use these methods to texture simple objects: a vase, for example, can be textured using cylindrical mapping, especially if you use a paint program to stretch and contract the image to correspond with the vase's curves. However, some objects are just too complex to be textured using projected mapping, which means having to resort to a fourth method, surface mapping. A surface map is generated when the object is actually constructed, and if you think of the object as having a skin, the shape of the map is the shape of that skin carefully peeled off and laid flat.

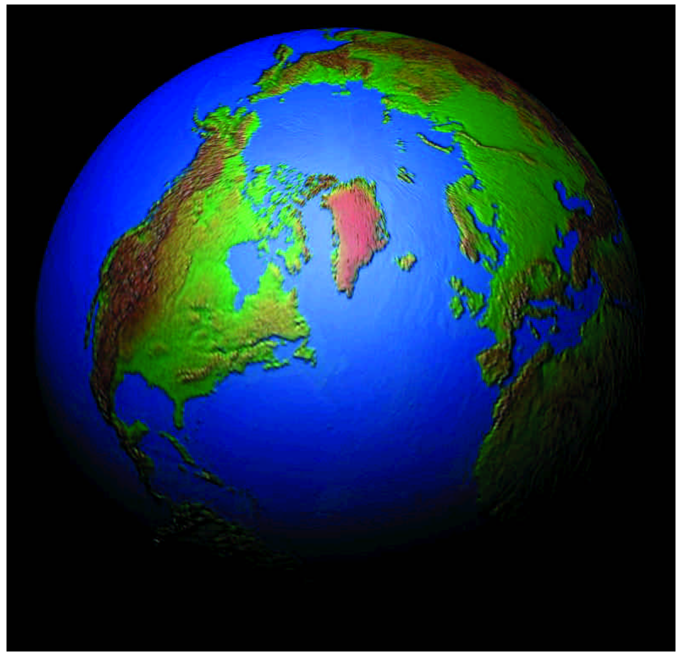


Fig 2 The texture map in Fig 1 wrapped round a sphere. Greenland assumes its proper proportions

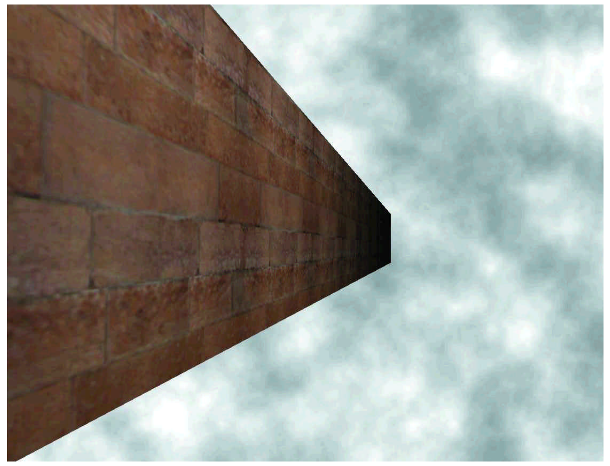


Fig 3 The purpose of this rather surreal image is to show a texture map being stretched beyond its resolution. Note the blurring where the wall is closest to our point of view

If you are having problems getting a surface map to work, a weirdly distributed surface map could well be the cause. One way of solving it is to create a texture covered with a grid, using a gradation of colours so you can distinguish the position of the lines. Apply this grid as a surface-mapped texture to the object and see if that throws any light on how the map is arranged. Another easier solution is, of course, being able to paint and stick textures directly onto the surface of objects without bothering about technicalities like mapping co-ordinates. Which brings me on to Fractal Design's Detailer.

Detailer

When I first read the blurb about Detailer, I could barely believe it. "Amazing 3D Paint Program" proclaimed the press release. "A stunning new graphics application that allows users to paint on the surface of 3D models in real time." This could be the answer to all my prayers, I thought; 3D painting on the PC platform.

After spending a few weeks with Detailer, I have to say that it only partially lives up to its promise. It *can* work in real time, but most PCs will be stretched to the limit to keep up. And the design is fussy, introducing a whole new set of terms and concepts to a field already overburdened with both. However, I should point out that even if it is not quite 3D painting in the full-blown sense, it does offer one crucial new capability: it brings 2D and 3D together.

Generally, when I am working with textures, I have a paint package like Photoshop and a 3D package open on the system simultaneously. I edit the image, save it, load it into the 3D package's texture editor, apply it and then render the object to see what has happened. When, as is inevitably the case, I find the texture is too big, too small, too bright, too dark, too whatever, I have to start again. With Detailer, these two functions are combined. You have one window showing the 3D model being textured, another showing the 2D texture. When you change the texture, you see the result immediately in the model window. And there is another facility that helps deal with the surface mapping problem: being able to overlay a "mesh" that shows in 2D the surface ("implicit" in Detailer parlance) map of the object being worked upon — the skin, if you will. You can then paint over the mesh, building up a texture that maps directly onto the surface of the object.

Fractal Design is an interesting and increasingly influential company in the graphics field. Painter 4, Ray Dream Designer, Poseur, and now Expression (my favourite: a program that allows you to use drawing tools to paint) make up a more than adequate toolkit for the budding computer graphics artist. Detailer will be a perfect complement to this developing suite once certain shortcomings are dealt with: when there is some sort of mechanism for importing surface/implicit mappings or, even better, deriving them from the geometry; when the interface and jargon is simplified; when you can export the flattened-out meshes of objects with implicit mapping so you can use more sophisticated 2D packages to paint over them. I hope this is not unreasonable. I only suggest it because Detailer so tantalisingly holds out the prospect of making texturing a simple, even intuitive process.

PCW Contact

Benjamin Woolley, writer and broadcaster, can be contacted at 3d@pcw.vnu.co.uk

All in the past

Or is it? Virtual heritage promises realistic experiences of Gettysburg, the Colosseum, Stonehenge, and Hitler's vision of a post-war Berlin. Benjamin Woolley steps back in time.

As I write, an event is already underway in the centre of London, carrying the intriguing title "Virtual Heritage 96". Virtual heritage? What could that possibly be?

To the snobbish, all "heritage" is virtual — a fake recreation of the past that panders to the public's poor knowledge of history. It's all about grand country houses opening up shops to sell tacky knick-knacks, sales executives dressing up as Roundheads, and theme-park rides through reconstructed peasant villages saturated with synthetic sewage smells. What could be less real, more virtual?

Stonehenge, for one. To demonstrate the power of its new generation of processors, Intel got together with English Heritage and, under the direction of Professor Robert Stone, the VR pioneer who now runs VR Solutions, created a VR version of Stonehenge that could be accessed over the internet using Superscape's Viscape browser, a proprietary client for viewing scenes generated using the company's VRT authoring software.

Thanks to the involvement of English Heritage (the quango that manages Stonehenge), the consortium was given access to the site and built up a precise database of its geometry. This database was then used to generate models that showed what the 'Henge would have looked like through the ages, from 10,500 years ago to the start of the next millennium. The result was a good demonstration of how VR (in the sense of real-time 3D graphics) can be used to recreate a lost past.

Although words like "photorealistic" were bandied about to describe the quality of the model, nobody could possibly be fooled into thinking that the images of the Virtual Stonehenge you saw in the Viscape window were photographs. Nevertheless, it did give you the vaguest notion of what it might be like to be there without the distractions of coachloads of tourists and carloads of screaming children. It let you get inside the ring of stones, something we have not been allowed to do in actuality for years.

That, then, is an example of "virtual heritage". To quote Dr William Mitchell of Manchester Metropolitan University, a speaker at Virtual Heritage 96, it "…gives users the freedom to explore monuments that may no longer exist or may have been damaged or spoilt by the effects of tourism. Exploring virtual reconstructions leaves no footprints and can potentially allow a user to examine details that are just not possible to see physically."

Dr Mitchell has himself contributed to our virtual heritage with a project entitled "The Tomb of Menna", which formed the basis of his contribution to the Virtual Heritage 96 conference. Menna was an Egyptian scribe of the 18th Dynasty (whenever that was) and his tomb was discovered earlier this century in Thebes, the ancient city across the Nile from modern-day Luxor. Its recreation has been achieved using VRML, and pretty impressive it is too (Fig 1). The geometry is simple. The detail lies in the textures, which are highly compressed JPEG images of the friezes on the tomb's walls.

This is just one of an expanding array of projects that make up the virtual heritage movement. There's virtual Gettysburg, a virtual Chinese Terracotta Army and the virtual Colosseum. I myself was involved in realising a virtual Germania. Hitler planned to rename Berlin as Germania after he had defeated the allies. He even got his architect, Albert Speer, to draw up detailed plans, which featured on a TV programme about Berlin's history and future as a united Germany's capital. With the help of our friends at the modelling company, Modelbox, we used the plans to render up a series of animations. It was, I remember, an exciting process, since it allowed us to experience the impressive and oppressive scale of Speer's grandiose vision in a way Hitler himself never could (Fig 2).

It is possible that virtual heritage is a passing fad. The point of preserving Stonehenge is to provide a means of keeping in contact with something authentic in an increasingly artificial world. So to that extent it seems to be a contradiction in terms. It might also provide an excuse for authorities such as English Heritage to deny access to monuments that are currently open to the public (tourist-free sites are, after all, a lot easier and cheaper to manage).

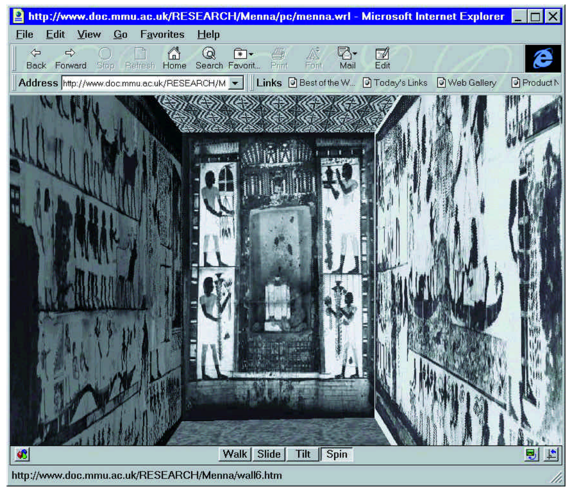


Fig 1 The Tomb of Menna by Dr William Mitchell

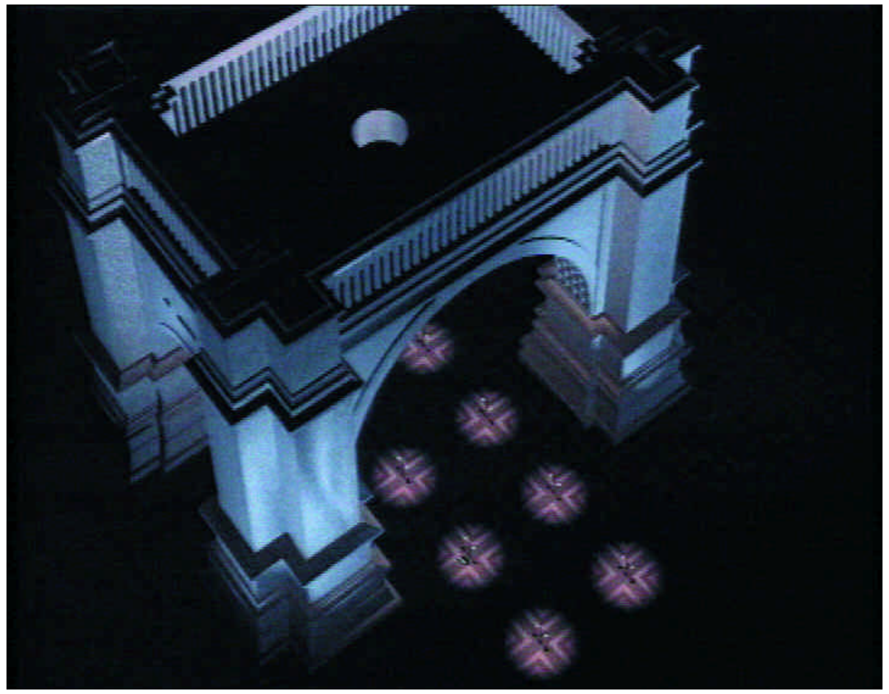


Fig 2 The Great Triumphal Arch of Speer's Berlin at night. This is a video grab, courtesy of Modelbox, hence the slightly fuzzy quality. It comes from the film we made recreating Speer's vision for Hitler's dream city. To give a sense of the arch's size, we inserted footage of a real car driving beneath it. At this scale it is a mere speck, barely visible at all, caught in one of the pools of street light illuminating the ground

But as IBM, for instance, demonstrated in its reconstruction of Dresden's magnificent Frauenkirche, which was demolished by the Allied bombing raids in World War II, virtual heritage provides us all with a valuable way of recovering what we can no longer experience.

The virtual universe's Big Bang

The technologies being developed for building shared spaces or multi-user virtual worlds or whatever you choose to call them are now emerging thick and fast. I am pleased to report that everyone is being extremely co-operative in this enterprise, even now that we have a new contender on the scene: Open Community, from the Mitsubishi Electric Research Laboratory (MERL) in Cambridge, Massachusetts (formerly known as Universal Worlds).

Open Community (www.merl.com/opencom/opencom.htm) does not come from the VRML community (although it will support worlds built using VRML models). Rather, it has its origins in a technology developed internally by MERL called SPLINE (Scalable Platform for Large Interactive Networked Environments). SPLINE has been under development for more than three years. Using it, a virtual world has already been built: Diamond Park, a place "where avatars could travel around a large park, talk to others using proximity-based voice chat, ride bicycles in a velodrome, create new world views, and play multi-user games".

To create Open Community, SPLINE has been combined with the Universal Avatar initiative (www.chaco.com/community/avatar.html) which aims to provide a standard for avatars so a virtual identity you create for one shared space on the internet could be used in another. The result is a sophisticated-looking application program interface (yes, yet another API) based on Java (yes, yet more Java) that embraces both the network and content sides of social spaces.

It is the fact that Open Community deals with the network side of the social spaces issue which, in my opinion, makes it particularly important because it is the network that makes social spaces unique, and presents the biggest challenge to making them work. The main problem is "latency". As we all know, you don't always get what you want from the internet when you want it. Data floods down the line one minute and dribbles down it the next. The reason for this is that the TCP/IP protocols, on which the internet depends, were not designed to deliver data in real time. They were designed to route things like email, files and scientific data which, generally speaking, one can afford to receive a minute or two later than expected.

For real-time applications, though, latency is a killer and shared spaces are, by their nature, real time. So Open Community promises to provide a set of tools which will manage this problem. A variety of techniques are suggested, ranging from the obvious (supplying the bulk of the data for a world on CD-ROM) to the ingenious. An example of the latter, given by the authors, is a simulated baseball game. When the batsman hits the ball, and a fielder runs to catch it, the batsman's "client" (the program running on the computer owned by the person controlling the batsman) anticipates where the ball will land, and passes on that information to the fielder's client before the ball has actually been hit. So the fielder's client can show the ball's initial direction even if the information about its actual trajectory is delayed by the network.

As we continue through 1997, I think the collaborative spirit in which Open Community and other initiatives are being discussed means there is a good chance of the industry doing justice to this most significant and exciting of 3D graphics/virtual reality applications. It is nice to start the year on such a positive note.

Render unto Criterion…

In the December issue column, I wrote about the Direct3D and QuickDraw 3D APIs. Who, I and many others were asking, will lead: Microsoft? Apple? Well, as I should have mentioned, for the time being neither will because the real leader is probably Criterion, the British company responsible for the RenderWare API. Criterion, now owned by Canon, claims RenderWare is the market leader. It is certainly popular, and is used in many games and VRML browsers such as Netscape's Live3D and SGI's Cosmo Player. It is fast, too (unlike Direct3D version 2, according to recent reports); you can see for yourself by trying out World Inc's AlphaWorld (www.worlds.net).

PCW Contact

Benjamin Woolley, writer and broadcaster, can be contacted at **3d@pcw.vnu.co.uk**

Out of this world

The virtual world is huge, and getting better all the time. Benjamin Woolley dons his avatar and goes on tour to produce a rough guide to strange lands.

I know the web is supposed to be a revolutionary new medium, different from all its predecessors, being interactive, using multimedia and all that. But when you think about it, most of the information you get is not so radically different to what you glean from print and TV media: flat pages of illustrated text that look like magazine pages, combinations of sounds, text and video that could pass for designer news bulletins. There is, however, one "media type" the internet can deliver which is really novel: the shared virtual world. By this, I mean a computer-generated space that a number of people can access simultaneously across a network and inhabit via a virtual stand-in or "avatar".

Experimental versions of such worlds already exist: notably the WorldsAway game which you can access through CompuServe, and AlphaWorld from Worlds Inc., which is on the net at www.worlds.net.

WorldsAway is not really a shared "space", since the environment is generated not out of proper 3D models but 2D backdrops upon which avatars and objects are superimposed. AlphaWorlds, by contrast, is more like the authentic article, and one that has been quietly developing a substantial 3D presence since its public launch in October 1995. It was created by Worlds Inc., to showcase the company's interactive 3D technology which it has dubbed, picking up on Microsoft's flavour of the month, Active Worlds.

Last October, the company announced that it would begin shipping an Active Worlds Development Kit (to run on Sun, SGI and Windows NT platforms) so that third parties can create and publish shared spaces of their own.

AlphaWorld is impressive. You access it by downloading Worlds Inc's own client or browser program and "teleporting" to the AlphaWorld co-ordinates. The first time you enter, you are confronted with a void. Slowly, the world takes shape before your eyes, object by object, texture by texture, efficiently "streamed" down the line so you (or rather, your avatar) can begin to wander around (using the mouse or cursor keys) before all the data has been downloaded.

The world is huge and getting better: the full data set for all the models and textures probably runs into tens of megabytes. Thankfully, data is cached to your local hard drive so the more you access the world, the faster it appears on your screen.

Some of the first objects to appear are avatars, represented by virtual mannequins of various sizes, shapes, sexes, species and demeanours. Each one you see will be driven by another person who is sharing the space. They can see you, just as you can see them, and you can interact with them in much the same manner as a text-based MUD, through gestures or "speech" (typed text, displayed as a speech bubble above your head).

When you apply for "immigration" to AlphaWorld, you are given a standard avatar, but you can select another from a whole library of character types, each identified by a suggestive name. For instance: Butch, Helmut, or Shred (the surfer) which is a particularly popular choice, as you can tell from Fig 1; two Shreds are walking past me as I stand in the middle of AlphaWorld.

Another, perhaps more interesting, form of interaction possible in AlphaWorld is being able to shape the environment itself. You can build on any unused section of property by duplicating objects you find elsewhere in the world and dragging them

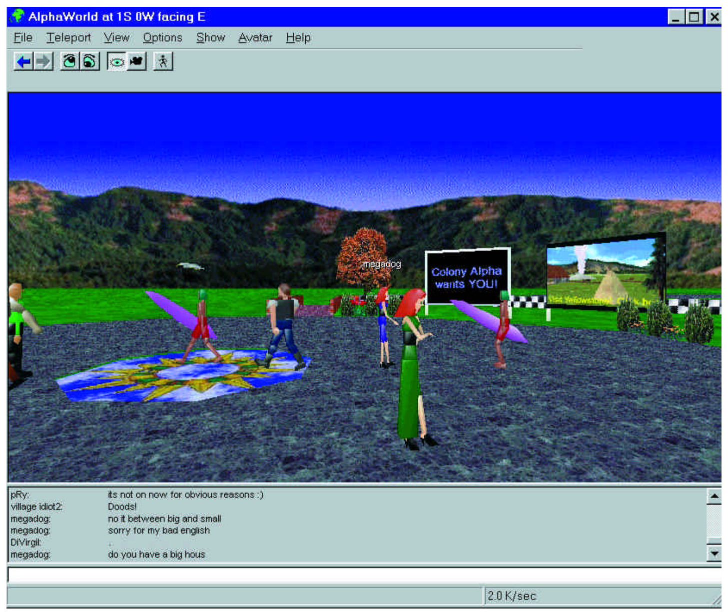


Fig 1 AlphaWorld's crowded central plaza

to your patch. You can alter some features of these objects (although not the basic geometry and look) and even give them behavioural characteristics. For example, your object could play a tune when someone bumps into it.

At the time of writing, a wide assortment of blue chip companies and other organisations were experimenting with Active Worlds technology and building their own spaces for people to explore. These include Visa which is designing a 3D online bank, Yellowstone National Park, and the Nokia phone company which is aiming to bring a little of the Scandinavian spirit to your screens.

One world which I considered to be particularly good was the Cyborg Nation (Fig 2). It was still under construction when I visited, and sparsely populated but given that what you see is being rendered in real time, I think it looks lovely. The sky and background are beautifully realised, and it is a delight to wander aimlessly around, awaiting some new object to spring up before you. I encountered the facade of a terraced house, a hovering metallic doughnut, a room with golden walls and a wireframe dome — it was rather like being in a Dadaist painting.

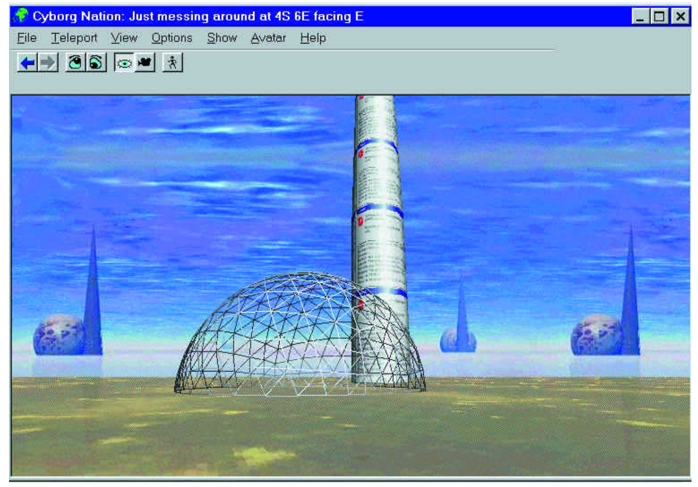
Although Active Worlds uses standard data file formats (such as RenderWare's RWX), which means third-party tools can be used to develop content, the system is proprietary. You will need the Development Kit to assemble worlds and publish them. This strategy has resulted in the steady evolution of an extremely effective product, but one that cannot rely for its future development on the same level of collaboration and competition as a technology relying on open standards. For that to happen, another approach is called for — one that is embodied in the new Living Worlds proposal.

Living Worlds

The idea behind Living Worlds is to use VRML 2.0 (see *Hands On 3D Graphics*, *PCW*, Dec'96) as the basis of a standard that allows the creation of the same type of shared virtual spaces which the Active Worlds technology already provides, but can be built, published and accessed using VRML-compliant tools and browsers.

Like HTML, VRML is totally public. Anyone can use it to create 3D objects and scenes that can be distributed across the web. Unfortunately, although it does allow

Fig 2 Cyborg Nation's virtual surrealism



the building of avatars and interaction with virtual spaces, these mechanisms are not standardised in a way that ensures true "interoperability", to use the term adopted by the Living Worlds team (a consortium of representatives from Sony, Paragraph, Worlds Inc., and others).

To illustrate the problem, the team dreamt up a series of scenarios: suppose someone called Art is at "home", suggest the Living Worlds team (in other words their avatar, or virtual presence, is in a 3D model of a living room realised using VRML); Art has recently "redecorated" his room, and there is new artwork on the walls that is automatically updated each month from some sort of interior design server.

This scenario shows how even the simplest of virtual spaces can quickly blur the distinction between authoring and using, and can come to rely on a variety of different sources and developers which update it, dynamically.

The Living Worlds team then imagine Art has some virtual visitors called Betty and Chuck (very American). They knock on the door. He opens it, sees them and greets them. This is the first point when some of the key interoperability questions are raised. How do Betty and Chuck "find" Art and how do they interact with him. Remember, there is no standard mechanism under VRML for words or gestures. Can they speak to each other, gesture, touch, sniff, hit… *mate*? — none of these points are unambiguously dealt with by VRML 2.0.

There are other, more subtle, issues the Living Worlds authors consider. What if you were able to exchange or buy virtual objects with behaviour characteristics? Suppose such objects could be delivered to you as complimentary gifts. What if the object were able to do some damage to your scene (perhaps a virtual puppy that bounces around Art's room, ruining the furniture and staining the carpet)? As the authors put it: "If this is beginning to sound like a virus, we've made our point. Multi-user apps in VRML, like those in any other language, will need some reliable way to protect themselves from inappropriate access."

Living Worlds is already coming up with answers to these questions, and in particular to the issue of avatars. There has already been an attempt by one team to create a "Universal Avatar" standard (you can find their discussion paper at www.chaco.com/avatar/avatar.html), and Living Worlds takes this a step further by refining the definition of an avatar and distinguishing it from other types of objects that would be expected to populate a shared space.

Avatars are usually defined as "transient and arbitrarily mobile" objects because they come and go, and are driven by humans. In contrast, other objects are "persistent and predictable" because they are driven by programs. However, most expect shared spaces to be populated by "bots" which are, essentially, program-driven objects designed to behave as if they were avatars, so any future standard will have to embrace their behaviour, too.

These are early days for shared spaces and the technologies that will create them. It remains to be seen whether it will be the proprietary approach (via Active Worlds and any emerging competitors) or the open standards approach (via Living Worlds) that will set the agenda and deliver the goods. Either way, it must surely be the area where 3D and the internet can create something truly unique.

PCW Contacts

Benjamin Woolley can be contacted at **3d@pcw.vnu.co.uk** He presents The Net, which will be broadcast on BBC2 from mid-January.

Active Worlds www.worlds.net
Living Worlds www.livingworlds.com

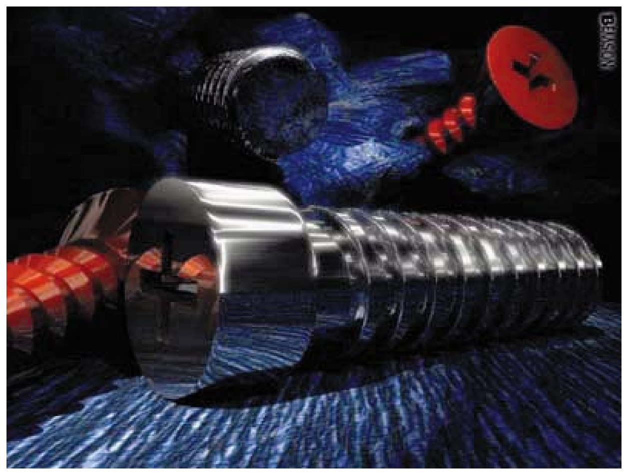
Direct action

Benjamin Woolley assesses Direct3D, Microsoft's promising API for adding 3D functions to applications. He also finds himself in the thrall of power mania: which hardware is big enough?

The internet is not the only area of the information revolution that Microsoft once neglected and is now determined to dominate. 3D graphics are also now firmly in the company's laser-guided sights. Its strategy has been to buy up existing technologies and Microsoften them up for global exploitation. One of these is Reality Lab, a set of programming tools originally developed by the British company, RenderMorphics, for rendering simple textured shapes in real time. Microsoft has renamed it Direct3D and developed it as the 3D component of its burgeoning multimedia application programming interface, DirectX, version 3 of which had just been launched at the time of writing this. The DirectX "evangelists" (a troop of which are bound for Europe, I am told) are promising that their technology will enable PCs to equal the current performance of consoles and arcades once hardware developments like Intel's MMX and Microsoft's own Talisman (see the November column) are commonly available.

Direct3D is an API, which means it acts as a sort of programming language (used in combination with an existing one, such as C++) for adding 3D functions to applications. Those applications may be games, they may be programs for authoring games and other 3D content, they may be molecular modelling packages, even databases or spreadsheets.

Microsoft has put a lot of work into Direct3D, and the demonstrations I have seen on the developer CD-ROM are promising. A simple textured sphere or teapot (the standard artefact for graphics demos), for example, will render smoothly in real time in a 320x240 window on a standard Pentium system. Direct3D (and, indeed, DirectX as a whole) also has the important feature of being able to take advantage of whatever hardware resources are available. If it finds a 3D graphics accelerator, it will be used, so long as there is a driver, which is likely, as most of the major 3D graphics chips are designed to support Direct3D. But equally important, if no acceleration is available, Direct3D objects, and any sounds or 2D images with which they are combined, will still be displayed, generated by a "Hardware Emulation Layer" that reproduces in software any functions that are unavailable in hardware.



An image entitled "Screw the Mold" by Sandford Bemi Faisonat, which features in the Apple QuickDraw 3D Gallery *(quickdraw3d.apple.com)*. The image was rendered using QuickDraw 3D, although obviously not in real time

Direct3D is not the only 3D API on the market. There is OpenGL too, which is aimed at the higher-end market and is already well-established. More significantly from a PC point of view, there is QuickDraw 3D from Microsoft's old rival, Apple. In at least one head-to-head comparison (published in the American magazine, *Byte*) QuickDraw 3D came out ahead of Direct3D for offering a greater range of object primitives and for its support of both the Mac and Windows platforms. Some of the Microsoft literature claims that DirectX, too, will be cross-platform. There are some doubts about this. According to at least one source within the company, the main purpose of the technology is to give 32-bit Windows operating environments a competitive edge over rivals, which obviously include Macintosh.

So which API will prevail, and does it matter? It certainly matters, because either Direct3D or QuickDraw 3D are likely to provide the basis for 3D becoming a standard part of the PC environment, as commonplace as sound and 2D graphics are now. You will need to consider this when choosing both software tools and hardware, trying wherever possible to keep your options open by getting support for both (which most third-party developers are, so far, promising to provide).

The question as to which API is likely to prevail is a trickier proposition. We all know who has the marketing muscle. We all know who controls the operating system (or at least, the one used by the vast majority). But 3D is a relatively new field in PC terms, QuickDraw is already well supported, and you only have to visit Apple's QuickDraw server (quickdraw3d.apple.com) to see that the company means business.

Power mad

Last month, my Compaq Deskpro's hard disk drive decided to experience a strange, slow-motion crash, deteriorating from full working order to complete cabbage-like coma in the space of an hour. I packed it off to my supplier, where it gathered dust for three weeks awaiting Compaq's delivery of a replacement.

During its absence I had to resort to my backup system, an old 486 Viglen Genie, which, unlike the Compaq, has chuntered away reliably in the background without a squeak of protest since I bought it some time in the last century. Being modestly specified in all departments except RAM (it has 16Mb), the Viglen, I thought, would prove to be unusable. In fact, I found it capable of doing just about the same amount of work. For 3D, I returned to Autodesk's 3D Studio running under DOS; for writing, Microsoft Word running under Windows 3.11; for the internet, good old Pegasus and WinFTP (I decided to forgo the delights of the web for a while). It was not tidy, it was not integrated, but it did work.

Those of us who are working with 3D graphics are currently in the thrall of power mania. We are constantly told that more means more: more processing power, more RAM, and yet more sophisticated software means more creativity, more spectacular effects and yet more moolah.

So when I sat down in front of what to many must still represent the pinnacle of desktop computing power, a Silicon Graphics workstation, it was in a mood of extreme scepticism. The machine in question was SGI's new "personal" workstation, the O2*. SGI's definition of personal is somewhat different to, say, Viglen's. The cheapest O2 costs just over £5,000, for which you get a 32Mb system armed with a MIPS R5000 RISC processor running at 180MHz. It offers blistering graphics performance through a "unified memory architecture" (i.e. no special-texture RAM) combined with built-in hardware acceleration and a system bus that can shove data around at a rate of 2.1 gigabytes per second.

I spent about an hour on the O2, and found it (temporarily at least) restored my confidence in technology. It was the first time I had used VRML that was both nice to look at and explore, smoother than anything I have so far experienced on an NT box or, for that matter, a Unix one. That, of course, was partly because it used the latest version of SGI's Webspace VRML 2.0 browser. But it also seemed to indicate that SGI might still retain the edge when it comes to optimising hardware for graphics.

However, do 3D artists have to start contemplating spending more than £5,000 in order to do decent work? Do we really need all that extra power? And if we do, should we pay the premium that is inevitable if you leave the general-purpose PC architecture behind and choose something from SGI? Or should we start thinking about going back to basics: stepping off the technology roller coaster, settling back with the old products that we know and like, and leaving it at that?

For me, for the moment, not even the allure of an O2 can completely discredit the latter strategy. But then, my Deskpro is now back and apparently working well, I have started to use the Workstation edition of Windows NT 4.0, and I have been eyeing a rather nice accelerator board. It can only be a matter of time before power madness once again prevails.

** See PCW December 96 for a full review of the Silicon Graphics O2.*

Infobyte

An Italian company called Infobyte specialises in creating VR tours of historical sites that are truly spectacular. They include the stunning Giotto frescoes in the Basilica of St Francis in Assisi (pictured here), St Peter's Basilica, the Coliseum and, most recently, the restored tomb of the Egyptian queen Nefertari, a VRML version of which can be explored by pointing your browser (running on an extremely powerful workstation) at the company's excellent web site, www.infobyte.it. Once this sort of thing runs in real time on an ordinary PC over a standard internet link, I think 3D's day will truly have arrived.

Dear Santa...

In its opening months, 1996 seemed it might turn out to be the moment when 3D finally fulfilled its promise. Creative Labs was selling the 3DBlaster board, VRML was becoming better known. However, the 3DBlaster did not turn out to be the graphics equivalent of the SoundBlaster because only individual programs (games) could take advantage of it. VRML, too, was a bit of a damp squib; few had the hardware to do anything with it, fewer the desire to spend their online hours wandering terrains that look like they were designed by the Early Learning Centre.

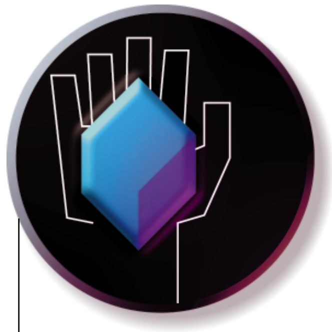
Now, as the New Year arrives, one gets the distinct impression that things are starting to move. With the Millennium board and now the Mystique, Matrox has started to establish 3D acceleration as a standard part of the PC graphics subsystem. With the plummeting price of memory, systems are coming equipped with the 16Mb of RAM that is the absolute minimum for handling textured 3D data. With the emergence of mainstream APIs (see main story), we at last have a mechanism for bringing the benefits of the third dimension not just to games, but to a whole welter of applications.

But I do not expect 1997 to be year zero: we have some way to go yet. Santa keeps forgetting to pack his sleigh with such essentials as modular, easy-to-use 3D authoring tools (the current crop are overweight and monolithic), a standard for plug-ins, and the imagination booster all of us involved in the graphics business need if we are to start to come up with content that is both wonderful and practical .

For me, what 1996 lacked most was a Myst, some game or virtual artefact that aroused one's excitement in the possibilities of 3D. So, Santa, please could you give us another of those in 1997? Not Myst 2, but something that demonstrates what wonderful, colourful, inspirational landscapes that even a humble PC can help create.

•PCW Contacts

Benjamin Woolley, writer and broadcaster, can be contacted at **woolley@illumin.co.uk**. His home page is **www.illumin.co.uk/woolley**



World vision

Benjamin Woolley takes a fresh look at VRML 2.0 — using Moving Worlds technology, it lets you use 3D models and simulations on the net.

When I last wrote about VRML 2.0 (the second version of the standard for using 3D models and simulations over the internet), I got into a spot of bother. I described the decision by SGI and Netscape to declare the Moving Worlds technology as the best basis for the new standard as "pre-emptive".

The two companies had announced the support of just about the entire 3D industry (excluding Apple, who is now on-side, and Microsoft) before there had been much discussion about the alternatives. My remarks were posted to the VRML newsgroup, where they provoked some sharp criticism (and a little support which was, for some reason, offered anonymously). It seems that a few members of the group did not like the suggestion that the VRML community was being, or could be, manipulated. Each proposal for VRML 2.0 would be assessed on its technical merits alone.

In the event, Moving Worlds was voted in as the new VRML standard by VAG, the presiding VRML Architecture Group. Microsoft's ActiveVRML, Moving Worlds' main contender in a field of six proposals, attracted a large negative vote. Obviously, VAG members felt that Microsoft's hold over the internet would develop very nicely without help from them.

Since the final draft of the standard was formally adopted on 4th August, VRML 2.0 has had a chance to get a toehold on the web, helped by a substantial presence at this year's Siggraph show in New Orleans. So this month I thought I would put all past disagreements behind me and have another, more thorough look at VRML 2, its capabilities and its future.

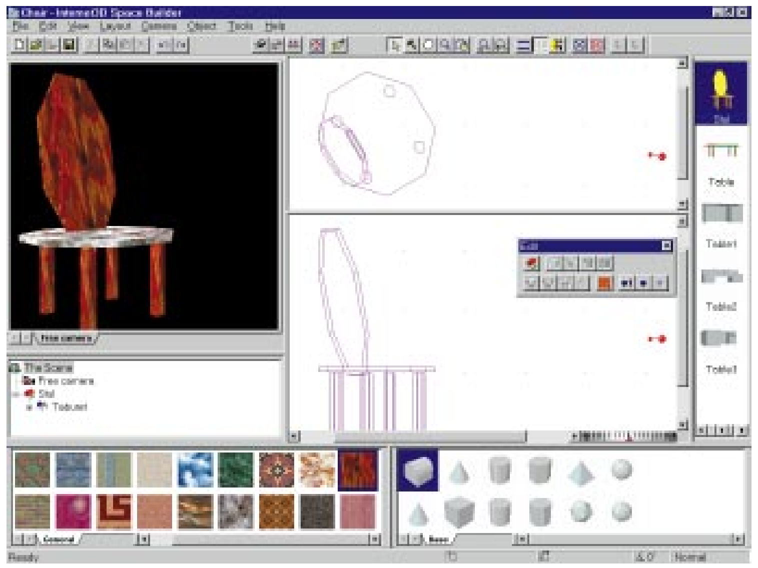


Fig 1 Paragraph's Internet 3D Space Builder

A better world

The standard is ambitious, promising to provide "a richer, more exciting, more interactive user experience than is possible within the static boundaries of VRML 1.0." There are five major areas of improvement: "enhanced" static worlds, interaction, animation, scripting and prototyping.

The enhancements to static worlds include the ability to put in backdrops, fog and bumpy terrain. Rendering scenes with fog may turn out to be too processor-intensive for all but the most powerful systems so, for a while, I don't expect to see it used much. But the backdrop facility which places a bitmap, such as a landscape, into the scene's background could prove to be a useful and simple way of adding a little more colour and character to a world.

The key to the interaction improvement is the concept of the "sensor". There are various "geometric" sensors that are triggered by events in space, and a sensor triggered by events in time. When a sensor is triggered, it can invoke some other node to be executed (a node is the VRML term for a programming command — see the *PCW* May edition of this column). An example of a geometric sensor is the ProximitySensor node. For this, you define a box-shaped region in space. If the user enters this space while navigating through the world, an event is triggered (for instance, an object in the vicinity becomes animated).

A particularly important enhancement to the VRML sensory environment is the introduction of sophisticated collision detection. A collision node prevents the user, or more precisely their "avatar", entering either specified geometry or all geometry in the scene. In particular, you can ensure that the user does not plough into uneven terrain instead of walking over the top of it.

In the field of animation, the third area of VRML 2.0 improvements, there is a whole new class of nodes called "interpolators", which can be used to alter an object's colour, position, orientation and size as well as other features.

Objects can also be controlled and given behaviour using scripting, the fourth main improvement. Interestingly, the VRML 2.0 specification does not specify which programming language should be used for scripting. The standard specifies that the language is one supported by the browser being used to view the world. This, of course, currently means Java but perhaps one day a developer will come up with an alternative that is tailored to 3D animation and simulation?

The final area of improvement is "prototyping", also known as "extensibility". VRML 2.0 allows new nodes to be created out of existing groups. You imagine this will typically be used to create nodes for complex objects. Since it is possible to pass parameters and event information to and from these prototype nodes, they can be controlled just like any other.

These and other enhancements have turned VRML from a basic 3D scene description language into a sophisticated animation and simulation programming tool. This should mean that, as promised, it can provide "a richer, more exciting, more interactive user experience". But in addition, it could also mean that creating these user experiences will be much more of a complex business. The VRML specification contains concepts and jargon that all but the most competent programmer will find daunting.

None of this is the fault of the standard's designers. They have tried very hard to make the full specification accessible and understandable. You can download it from the VRML home page (vag.vrml.org). It is a 1.5Mb file that has been compressed using the Unix "tar" format and you will need a utility like WinZip version 6.1 to decompress it. You will find it very well laid-out in HTML format, with convenient links for jumping between the various sections. There are a few tutorials, one based on a Siggraph '96 session (at www.sdsc.edu/siggraph96vrml/) and a couple available through SGI's VRML server (vrml.sgi.com/experts/vrml2tutorials.html). Of course, by the time you read this, there may be more.

For those with neither the time nor inclination to tackle such complexities, it is worth trying out some authoring tools that are beginning to emerge. You should discover a list of those online at the starting point for all VRML work, the VRML Repository at /www.sdsc.edu/vrml/. At the time of writing, only two tools were listed that supported VRML 2.0: Internet 3D Space Builder and Virtual Home Space Builder, both from Paragraph.

I tried Internet 3D Space Builder (Fig 1) and can report that it is one of the neatest, nicest 3D apps I have yet downloaded from the internet (from Paragraph's web site at www.paragraph.com). I was using a beta version that had no documentation and did not support all of VRML 2.0's features like animation but in its basic design it worked like a dream, allowing me to build worlds out of primitives and a small collection of more complex objects (mostly office furniture) simply by dragging them into the scene. A preview window showed what the result would look like in a browser and even allowed me to drag textures straight onto the surfaces of objects.

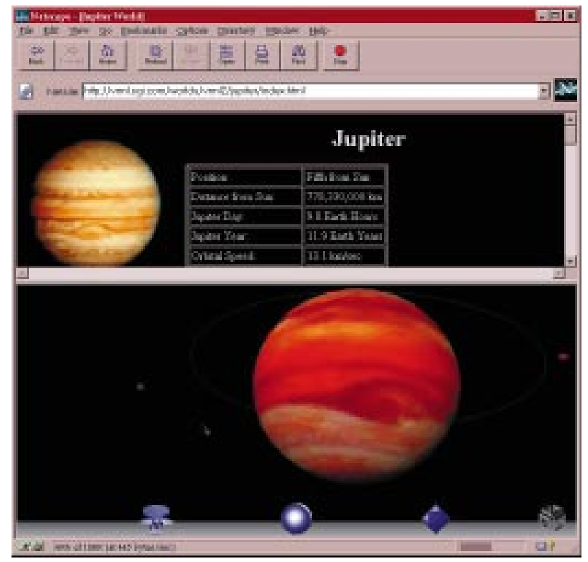


Fig 2 SGI's Jupiter demo: the lower panel in the window provides a 3D tour of Jupiter and its moons, while the top panel delivers the facts

It's a small world

Before having a go at building your own world, you might want to see what others have achieved so far. At the time of writing, there was little to see. Unlike Java or Shockwave, you sense a reluctance among content providers to use VRML and you can understand why.

The hardware is not yet in place to make realtime 3D a credible form of communication. Many people still have 486s, most have 8Mb of RAM or less and hardly any have 3D acceleration, or connections faster than 28.8bps. This means that the simplest VRML world or object behaves as if its batteries had run down. VRML tends to look ugly, as well, because the detailed textures are too heavy on resources.

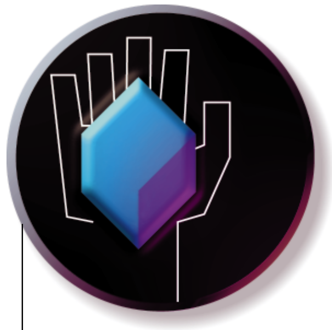
Nevertheless, there are a few demonstration worlds around (see the Jupiter example in Fig 2), hinting at the riches to come. I viewed them using the beta 2 version of SGI's CosmoPlayer, which was, at the time, one of only three browsers listed by the VRML Repository as supporting VRML 2.0 (Netscape's first version of Live3D is a subset of VRML 2.0). I also tried a couple of scenes created using a Doom-to-VRML 2.0 converter. The results were incredibly slow to load and run, but suggested one possible source of material that would look good once 3D accelerators become more commonplace.

There remains some debate about whether VRML is the way to go with 3D on the internet. Various companies are touting alternatives. According to the graphics industry newsletter *Wave*, there is growing interest in using the so-called "DIS-Lite" standard as an alternative. DIS (Distributed Interactive Simulation) is a protocol developed by the American Department of Defense for networked simulations of battlefield operations. The companies that are working in this field, like Mak (www.mak.com), see a lite version of DIS as being the most effective way of building up a new generation of internet simulations and games.

The VRML Architecture Group may have voted on its vision of the 3D future but I sense that when it comes to the wider industry, the jury is still out. The time when VRML enjoys the same sort of global acceptance as HTML, or the same level of commercial support as Java, will be like a complex 3D world downloading onto a 486 via a v34 modem — slow in coming.

•PCW Contacts

Benjamin Woolley, writer and broadcaster, can be contacted at **woolley@illumin.co.uk**. His home page is **www.illumin.co.uk/woolley/**



Why did the **chicken** cross the road?

To see Talisman, the new 3D hardware architecture from Microsoft. Benjamin Woolley looks at its application in real-time graphics, and gets in a twist about special FX.

I first visited SIGGRAPH, America's annual computer graphics megafest, in 1989 when it was held in Boston. I still have the mousemat to prove it, which features a large red lobster (Boston's unlikely choice of mascot). Even in those days, SIGGRAPH was huge, attracting upwards of 20,000 delegates from all four corners of the globe and the computing industry. It was there that I remember Al Gore, then a humble senator, now vice-president of the USA, opening the event with a live-by-satellite speech in which he talked of information "exploding in leaps and bounds". A wonderfully Moulinexed metaphor it may have been, but it accurately captured SIGGRAPH's transformation into one of the computing world's key events.

It was at SIGGRAPH 89 that people started talking excitedly about this newfangled virtual reality idea, and gazed with amazement and amusement at Jaron Lanier, the then fledgling VR industry's chief guru, doing his strange sort of jam session thing during what was otherwise supposed to be a serious technical conference.

These were the first tinglings of excitement that now seem to electrify SIGGRAPH every year as it becomes ever more firmly established as the venue for unveiling the most exciting ideas and developments in visual entertainment. At this year's conference, held in New Orleans, they came in their tens of thousands to get a peek at next year's movie effects, web content and games. Where Boston was full of bearded programmers and conceptual artists, New Orleans attracted the likes of Jeffrey Katzenberg, co-founder (with Stephen Spielberg and David Geffen) of the new computer-literate Hollywood studio, Dreamworks. It is also where you find a welter of new animation, including a strange little cartoon called "Chicken Crossing".

Finger-lickin' good...

Chicken Crossing was neither produced by DreamWorks nor any other studio. It came from Microsoft, a company that did not even attend Boston yet was out in full force in New Orleans. Although amusing enough as a work of entertainment, Microsoft's first attempt at a cartoon had the primary purpose of showing off "Talisman", a new technology being developed by the company's research division. This is, Microsoft states, "a new 3D graphics and multimedia hardware architecture" and if Chicken Crossing is anything to go by, it's the first sign that decent real-time 3D graphics may at last find their way onto the home PC.

First, let us consider what we mean by real-time graphics. In a 3D game like, say, Myst, or a movie with 3D graphic effects such as Twister (*see later*), the computer-generated images you see take hours, sometimes even days, to produce. So, obviously, they have to be done in advance. As a result, a game like Myst cannot strictly be 3D. Rather, it is a slideshow of 2D images with various puzzles determining the order in which they are seen.

A game like Doom is very different, because as you wander around those interminable tunnels (I am not a fan) the images are more or less generated from scratch as you go. This is necessary if the game is to allow you to roam freely through the artificial world it is trying to recreate, because to pre-render and store each possible scene as witnessed from every possible point of view would require impractical quantities of rendering time and storage capacity. Games like Doom deploy a special set of graphics tools (known as APIs) which use a variety of nifty shortcuts and compromises to generate each image as and when it is needed.

Fig 1 **A particle system in action**

Fig 2 Three stills from Chicken Crossing

Several APIs have been developed for this task, one of the best known of which, Reality Lab, was created by a British company called RenderMorphics. Like so many innovative British high-tech companies, RenderMorphics was snapped up by Microsoft which used Reality Lab as the basis for Direct3D, which itself is a subset of a whole library of APIs designed for multimedia content, called DirectX.

DirectX provides the software layer for the Talisman architecture, and Chicken Crossing was supposed to demonstrate what the two could achieve, in combination. According to Microsoft, a Pentium PC with Talisman hardware (which should only cost two or three hundred dollars) could render and display each of the frames you see in Fig 2 and the 6,997 that made up the rest of the Chicken Crossing animation, in the time it takes for the screen to refresh (in other words, around one 75th of a second). This is an astonishing claim, given the richness of the textures and the number of objects: way beyond anything currently achievable on a Pentium system, even one with hardware acceleration.

In an extremely technical paper presented to SIGGRAPH, Microsoft explained how this impressive trick could be pulled off. Talisman, like any graphics technology, works by making compromises, the most important of which is layering. Most 3D scenes are rendered as true three-dimensional spaces, with the shading of each element of the scene calculated according to its position and orientation with respect to the rendered point of view. Talisman instead associates particular objects in a scene with particular layers, and then decides how much work needs to go into rendering each layer. So, for example, a layer comprising an object disappearing into the distance needs very little render time at all. Indeed, the effect can be reproduced in a 2D rather than 3D scene by scaling down the 2D image of the object as it recedes.

On the face of it, this is a clever solution, although how smart Talisman-based software will be when it comes to deciding how to handle layers, remains to be seen.

Another compromise is one that sounds rather obvious, even low-tech. It is graphics compression. For various technical reasons to do with the way a scene is calculated, compression is difficult to achieve with conventional renderers. With Talisman, the scene is rendered in blocks 32 x 32 pixels in size (the process carries the unglamorous name of chunking), which can be compressed using the same sorts of techniques used by the JPEG graphics format.

Microsoft says it will not be making Talisman boards, but will license the detailed "reference" design to hardware manufacturers. The company claims that because the design of the silicon is relatively simple and because many of the main components will be standard parts, boards should retail for less than $300. If this is the case, then that really should set the cat among the crossing chickens.

Particles

Summer is about blockbusters, and nowadays blockbusters are about showing off the latest computer graphics effects. Some of the most impressive were to be found in Twister, a movie about tornadoes. In my opinion, the computer-generated tornadoes were the most realistic feature of the whole movie (far more realistic than the characters) and I began to wonder about how they might have been produced. With the help, it turned out, of the resources of Industrial Light and Magic, AliasWavefront, several very pricey plug-ins and about 20,000 lines of customised code.

Having returned home, I tried to brew up a tornado for myself. Naturally I failed (it looked like an upturned tree trunk) but I did manage a smoking chimney (Fig 1).

The key to such effects is a set of 3D tools called particle systems. These are not yet to be found in cheaper 3D packages but they should trickle down into future releases. There are a number you can buy as plug-ins for mid-range programs: for Lightwave, for instance, you can buy products like Particle Storm for about £300.

I used a 3D Studio Release 4 plug-in called "Vapor" to produce the smoking chimney. It is an unexceptional effect but, believe me, it was not easy to create. All particle systems make enormous demands on the processor, not least because being effects that develop over time, they have to be calculated for each frame of an animation. This means that until you render the animation, which can take ages, you cannot really judge whether you have correctly captured the dynamics of your smoke trail or twisting tornado.

The key to all particle effects is a special class of objects called "emitters". These emit a series of smaller objects (the particles) that are generated at a particular rate and disperse in a particular direction, in a particular formation, at a particular speed.

The Vapor plug-in comes with a series of presets for producing different types of smoke, from a cigarette trail to a steam locomotive's billowing clouds. The latter was not particularly convincing, so I had to fiddle around with the parameters to achieve the effect seen in Fig 1 (which, I hope you will appreciate, looks a lot better when animated). Each change to the size and intensity of such parameters (the "whorl" and "turbulence") produced rather unpredictable results, so it took a good few goes, and hours of rendering time, to tune the effect. It just goes to show that there is no smoke, and no tornado, without toil.

PCW Contacts

Benjamin Woolley, writer and broadcaster, can be contacted at **woolley@illumin.co.uk**. His home page is **www.illumin.co.uk/woolley/**

3D Graphics Hands On

Rays of light

POVRay is a lot of raytracing software downloadable in a 4Mb website file, and despite its heavy maths and programming bias, Benjamin Woolley got to grips with it.

As someone once said, “There is no such thing as a free lunch”. But there *is* such a thing as a free raytracing 3D graphics package. POVRay (Persistence Of Vision Raytracer) is among the most sophisticated around and, as I write this column, the Beta test phase of version 3 is drawing to a close. It is available in all flavours including DOS, Mac, Amiga, Unix, Linux and, as tested here, Windows. By the time you read this, the final version should be available, and I urge you to download it from the official POVRay website at www.povray.org or CompuServe’s GRAPHDEV forum.

The self-installing executable you get is over 4Mb, but this includes documentation and a generous helping of sample files and some fairly substantial binaries. It is quite remarkable that you can get such a lot of software in a file of this size, given that a commercial 3D package would come on a CD-ROM and colonise half your hard disk.

Installation is no more than a double-click on the downloaded file and a few simple answers to a few simple questions. By the time the disk-thrashing is over, you should find POVRay for Windows seamlessly settled into your Windows 3.11, 95 or NT system (it is a full 32-bit application) and ready to run.

I have to admit that the first time I used it, my feeling was one of disappointment. With POVRay, you re-enter a world that many of us had hoped to leave behind: the world of programming, command-line interpreters, declarations, variables and, (ugh!) maths. What follows is a sample taken from a tutorial in the help file for creating a colour gradient:

If you want to start one of the colours at a specific angle, you’ll first have to convert the angle to a colour map index. This is done by using the formula

```
color_map_index = (1 - cos(angle))
/ 2
```

where the angle is measured against the negated earth’s surface normal.

I have not personally encountered a cosine since fifth form, and can’t remember what one is, except that it has something to do with angles.

However, the mathematically timid should persist because, as I soon began to discover, the wonder of POVRay is that even without maths or a fondness for programming languages you can achieve a great deal.

Unlike commercial 3D graphics packages, POVRay is just a renderer. It does not include a modeller. This means that it takes scene descriptions, essentially 3D graphics programs, and turns them into rendered images. To use it, you have to write these scene descriptions yourself or use a program that will generate them for you.

I began by trying to write a few scenes for myself. There is a series of tutorials in the help file which helps you start coding, and you will find that you are soon able to create simple objects. The best way to proceed is to copy the lines of code supplied in the tutorial and paste them into a text editor. You can save the text as a file with the .pov extension, start up the POVRay renderer, and watch the scene emerge before your very eyes. If you have made a syntax error, POVRay reports which line caused the problem. By adjusting a few parameters here and there and re-rendering the scene, you can begin to get a feel for their effect.

An example of just about the simplest scene description file you can get is given in Fig 1. The “include” statement at the beginning merges the commands contained in the “colors.inc” file into the scene description. “Include” files can contain any legitimate POVRay statement, but are typically used to contain data, such as the definitions of colours, shapes and textures. They define Cyan as having the red/green/blue values 0, 1, 1 which means no red, full green and full blue. The rest of the program does as indicated in the comments prefixed with //.

Fig 1

```
#include “colors.inc”

// First create a background colour
  background { color Cyan }

//add a camera at position 0 units along the x
//axis, 2 units along the y axis, and -3 units
//along the z axis.  It points towards another
//point at co-ordinates 0,1,2, the position of the
//sphere
  camera {
    location <0, 2, -3>
    look_at  <0, 1,  2>
  }

//create a sphere, two units in diameter and
//colour it yellow
  sphere {
    <0, 1, 2>, 2
    texture {
      pigment { color Yellow }  //
    }
  }

//add a white light
  light_source { <2, 4, -3> color White}
```

Listing for a simple POVRay scene description

POVRay has an incredibly powerful scene description language that allows you to create 3D fractals, superquadric ellipsoids (which are objects with soft edges), halo effects, layered fog, and dust clouds. It creates more than those packages costing thousands of pounds. In its atmospheric capabilities, for example, it is ahead of 3D Studio Release 4. To exploit such features to the full, you should probably spend some time poring over the documentation, probably buy a book or two (for example, *Ray Tracing Worlds with POVRay* by Alexander Enzmann, Lutz Kretzschmar and Chris Young), and get the CD-ROM.

For the lazy ones among us, there is an easier way. You get an existing file and mess around with it. This is what I did to produce the image in Fig 2, which was rendered using POVRay version 3. I created it by adapting a 3D Studio file, which meant I could use 3D Studio’s modeller to work on the geometry. There are POVRay modellers available as shareware, the best-known of which is Moray. It is not the most wonderful piece of software and if anyone knows of anything better that is free or cheap, drop me a line. Meanwhile, I shall continue to look around for myself.

Fig 2 **Scene rendered using POVRay. The reflections show some of the advantages of raytracing over faster but cruder scanline renderers**

I converted the 3DS files generated by 3D Studio using a lovely freeware program called 3DS2POV, by Steve Anger and Jeff Bowermaster, downloaded from the CompuServe GRAPHDEV forum. Using a text editor, I adapted the resulting .pov file by borrowing little bits of extra code from the tutorials, to create the clouds in the background. I spent nearly all my time with POVRay using this jackdaw strategy, taking existing bits of code, playing around with them and rendering up the result to see what sort of mess I had made.

Since POVRay is freeware, widely distributed and designed to run on just about every type of computer you can think of, short of Babbage’s Analytical Engine, there are endless samples you can use and abuse in this manner. All samples are generously donated by their authors and widely posted across the internet and on various online services, mostly on CompuServe.

Night and Day

The image pictured right first appeared in *3D Artist*, the American magazine for 3D graphics users. I found it on the magazine’s web site, which is well worth a visit: www.3dartist.com.

The picture may be familiar to some of you. It is by Maurits Escher, the Dutch artist who for many expresses the weirdness and beauty of the information age with his mind-boggling images of infinite loops and distorted perspectives. People interested in the origins of his association with computing should look at Douglas Hofstadter’s wonderful book, *Godel, Escher and Bach*.

Escher’s best-known picture is probably “Ascending and Descending” (1960), which shows little elfin figures trooping up and down a staircase, the bottom of which impossibly connects to the top. Less well known is “High and Low”. Well, you see a version of it here, renamed “Night and Day”. It was created by Richard Stein III, a 3D artist who worked on the 7th Guest and 11th Hour games for Trilobyte, and who kindly gave me permission to reprint the picture.

Stein generated the image using 3D Studio. He points out, “An X-Y-Z-based program doesn’t have enough perspective points for us to build this type of image accurately. Morph software won’t do it. Stretching the camera lens beyond fish-eye won’t do it. But bending the objects in a specific way just might fake it.” And this is exactly what he has done, bending the objects away from the centre so that the illusion of Escher’s original picture is reproduced. The result is a fake in the sense that the illusion would quickly be lost if you tried to create an animation that moved through the scene. Because it was generated from a 3D model, Stein could render the scene as a “stereo pair”: two pictures showing the same scene from slightly different perspectives, thus giving the image real depth. I spent ages staring at the pair on my monitor, and succeeded in getting flashes of the stereoscopic effect. Escher would have loved it.

At the time of writing, Robert had put a range of his stereo pairs on the web. They are splendid, and you may still find them at www.tbyte.com/people/stein/stereo.htm. I have also reprinted the picture because, to me, it provides an object lesson in the effective use of materials: look at the floor in the centre of the image; the sheen of the stone is perfect. The walls have a rich, almost tactile texture to them. It just goes to show how wrong people are in thinking that computer-generated art is plastic-looking.

•PCW Contacts

Benjamin Woolley, writer and broadcaster, can be contacted at **woolley@illumin.co.uk**. His home page is **www.illumin.co.uk/woolley/**

It'll be all white on the night... won't it?

Benjamin Woolley **sets his tracing skills to work on the White Tower at the Tower of London. And the Intergraph made an impression — on his table.**

Here's a story about how to turn real buildings into 3D models, with nothing more than an obsolete scanner and a nifty graphics utility.

A colleague asked me to compile a short animation featuring the Tower of London. All he gave me to work with was a series of old CAD models of the Tower, and a couple of days to do it in. The CAD files were huge and used the AutoCAD DXF format.

It is technically quite difficult to convert to a 3D animation package file format, including Autodesk's PRJ/3DS format. I did manage to convert some of the files, but the level of detail was so high, as it tends to be in CAD models, that it would have taken days, possibly weeks, to identify each object, label it and texture it. Given the urgency of the job, I decided on a quicker and dirtier tack.

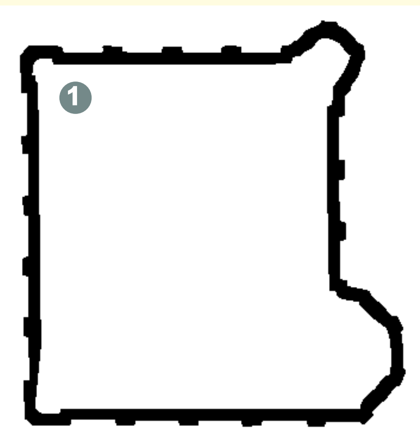
My starting point was a ground plan of the central White Tower in a book about the Tower of London's history. I scanned the plan using my trusty Logitech hand scanner, setting the colour depth to black-and-white to ensure maximum contrast between the outline and the background. I touched up the result using Photoshop, to cut out extraneous detail and scanning noise. You can see the finished image in *Fig 1*. It doesn't look exciting, but it was just what I needed for the next stage, namely tracing.

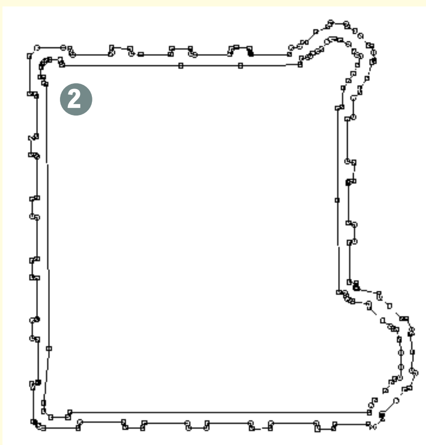
(3) *The result of lofting the traced image, with a stone material mapped on to its surface*

Tracing turns the flat bitmap image into a vector graphic, which is a drawing made up of lines and vertices, as shown in *Fig 2*. The software I used was the OCR-Trace component of CorelDraw 6. Having gone off Corel following my experiences with CorelDraw 4, which was prone to crashing at vital moments, I feel the company has redeemed itself with version 6. Not least because of the improved quality of all the ancillary software, OCR-Trace included. It did an excellent job.

I exported the vector graphic as a DXF file, which I could import into 3D Studio as a shape. I tidied up the geometry, and then "lofted" it into a 3D object *(Fig 3)*. This provided the basis of the finished model *(Fig 4)*. There's a lot wrong with it, not least the texture of the walls, which is the wrong colour. The building is not called the White Tower for nothing. At least I know the general shape of the architecture is accurate.

This scanning/tracing technique has a number of applications. I could photograph an object from a number of angles using a conventional camera, develop the picture on to Photo CD and use a bitmap editor to emphasise the edges. Most bitmap editors, such as Photoshop and Corel PhotoPaint, have this facility. Submitting the result to the tracing program would generate a series of profiles that could be used to build a 3D model of the original. It would be a lot of work, but cheaper than using a 3D scanner.





(1) *The scanned image of the White Tower's floor plan*
(2) *The traced image. The little squares mark the position of the vertices*

(4) *A render of the final model of the White Tower. A faithful rendition of the original, except that it is not white*

Balance of power

Like great empires, great operating systems rise and fall. Unix, some predict, is about to be toppled from its pre-eminent position as the world's industrial-strength workstation operating system. And Windows NT, we are told, will be its replacement. Unix users will scoff at such a suggestion. NT, they say, cannot cope with more than a handful of processors, and does not enjoy Unix's track record for running "mission critical" installations.

All of this may be true, but the balance of power between the two has never been more finely poised. The reason is the emergence of a number of pumped-up PCs offering workstation-class performance for a relatively modest price. I have been trying the Intergraph TDZ-300, and the combination is awesome.

Let's not pretend that the Intergraph is an ordinary desktop system. The model I was using featured a 200MHz Pentium Pro processor, 64Mb of RAM, 12Mb of VRAM on a card boasting Intergraph's own OpenGL 3D graphics acceleration, a monster 21in screen which nearly made my table collapse, and a 2Gb hard disk. That lot retails at about £14,000.

Starting up the Intergraph was a strange sensation. The bootup sequence featured the same series of BIOS messages you would find on the most humble PC. Even with Word or Excel running under Windows NT, it felt like using a MiG for a package flight to Spain. Only with Photoshop, Painter 4 and 3D Studio Max loaded did the full power of the hardware begin to manifest itself. My response was excitement tinged with disappointment.

The excitement comes from seeing what a Pentium Pro in a fast system can do. You do not need fancy benchmarks to observe the performance boost when you are doing 3D work. It is as though, until this moment, you have been working in a mud bath, with every movement and manipulation a laborious effort. With the Pro, reactions are instant. A rendered preview changes in real time, booleans happen in the blink of an eye, and models of complex architecture can be moved around the screen as though on a cushion of compressed air.

The disappointment comes with the discovery that, even with a 200MHz Pentium Pro under the bonnet and all that RAM and VRAM, the system has limits which are quickly reached. A polygon count running into the tens of thousands plunges you back in the mud.

This is to be expected. All workstations have their limits, even ones running Unix. The Intergraph, or even a top-of-the-range Dell or Compaq, in combination with NT, shows that those limits are no longer beyond the reach of the PC.

Max attack

The Intergraph gave me the opportunity to get my teeth into Autodesk's all-new 3D Studio Max, and I relished it. This month I want to dwell on one or two of Max's problems, not because it is bad, but because it is good. It is a package that pro and semi-pro 3D artists have to assess if they want to keep up with the state of their art.

The problems mostly concern the interface. Firstly, it is completely different from that of 3D Studio Release 4 (3DSR4). The learning curve required to move from 3DSR4 to Max is no gentler than the one you must climb to move to LightWave, which currently costs £2,000 less than Max. This is important to remember when working out which upgrade route to take.

The second problem with the interface concerns its aesthetics. Autodesk, or Kinetix, the company's new brand name for its 3D products, is proud of the look of Max, claiming it is all the things GUI interfaces are supposed to be: intuitive, simple and elegant. Compared to 3DSR4, it is all of these, but by the standards of modern Windows and Macintosh applications, it's a mess. There is simply too much of it exposed to the user at any one time.

Furthermore, it raises expectations of a level of interactivity that is not quite delivered. You cannot change the geometry of an object directly, except when you create it. You have to do so via a parameters panel, although changes are updated interactively, which almost substitutes.

Another problem is part of one of the product's greatest strengths: its modularity. To get the most out of Max, you will need plug-ins. Some come as standard. There are particle and "bones" systems which can be used to create falling snow or skeletons. Most users will need to buy non-standard plug-ins, and the cost will not be trivial. Character Studio, the Autodesk character animation plug-in, is priced at £600.

Modularity also means that compatibility could become an issue. A model that relies on non-standard plug-ins for its geometry or materials will only work on a Max system that has those plug-ins installed.

Such problems need to be set against the fact that Max is excellent. It's sophisticated and richly specified. Even after weeks with it, I am only beginning to scratch the surface. It makes good use of NT's multiprocessing capabilities, now boasts a truly exceptional, if quite complicated, modeller and materials editor, and has a renderer that makes a clever compromise between quality and time. It offers intriguing features like the ability to render over TCP/IP networks, which means, in theory, you could have render farms spread across the Internet. You also have good documentation, and the reassurance of knowing that you are a member of a user base that is likely to prove as extensive and supportive as 3DSR4's. ▣

PCW *Details*

Benjamin Woolley, writer and broadcaster, can be contacted at woolley@illumin.co.uk.
His home page is **www.illumin.co.uk/woolley/**

Model behaviour

Benjamin Woolley gets moving on the animation abilities of Autodesk's new 3D Studio Max and Fractal Design's Poser.

Within Autodesk, they called it the "AutoCAD burp". Every time AutoCAD (the world's most widely used CAD program) had a hiccup the whole company would shake. AutoCAD accounted for the majority of the Autodesk's income, pushing all the software publisher's other products and 3D Studio in particular, into second place.

Autodesk has now taken action to cure this bad case of corporate indigestion by setting up Kinetix (www.autodesk.com/kinetix), a division totally devoted to 3D graphics. The man in charge of Kinetix, Larry Crume, says the division's motto is: "We Be Three Dee". And to mark its birth, Kinetix has been given the job of launching one of the most eagerly awaited packages on the computer graphics market, 3D Studio MAX (*Fig 1*).

MAX has been a long time coming; so long that it was the subject of an April Fool's joke in the *Tessellation Times* (Tess for short), an American computer graphics publication. According to Tess, the arrival of MAX was delayed because Autodesk had been in dispute with the creators of the Max Headroom character, over the use of the name.

In fact, the delay had been caused by the usual problems of getting the code (and in particular the renderer) completed on schedule. The finished version finally started shipping at the beginning of May, which is when I got my copy, neatly packaged in a box bearing the Kinetix logo and accompanied by a document entitled "Reviewer's Guide" (designed, it appeared, to steer me towards looking at all the things the software is good at and away from all the things at which it is bad).

MAX has been eagerly awaited by users of the venerable DOS-based 3D Studio, which is beginning to show its age. it was also keenly awaited by the rest of the computer graphics world because of 3D Studio's undoubted impact on the market. The Reviewer's Guide informs me that 3D Studio is "the world's best selling professional 3D rendering and animation system with over 65,000 installations". This might be true. AutoCAD has helped beat a path to the doors of countless CAD departments around the world, in need of a 3D package to help them visualise and demonstrate their designs.

MAX is not, the company keeps saying, a new version of 3D Studio (which is still shipping as Release 4). It is a complete rewrite. It runs natively under Windows NT rather than DOS. (It will run under Windows 95, but you get a warning when you run Setup that it may not run properly — this proved to be the case when I tried it). Autodesk will only support NT.

It is structurally different, too, as it uses object-based programming techniques. All the tools in MAX are programming objects — DLLs that plug straight into the operating environment. Every component can be replaced or enhanced, even the renderer.

Fig 2 *A sample image supplied by Kinetix to show off MAX's new renderer. By old 3D Studio standards, the sharpness of the highlights combined with the softness of the atmosphere is good*

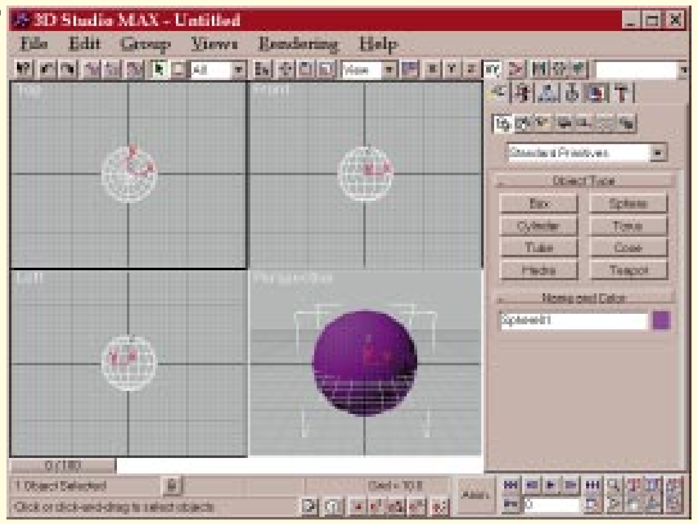


Fig 1 *The MAX interface is "modeless": you do not have to swap between different modes when performing different functions such as animating or modelling*

The renderer supplied is supposed to be an improvement on the rather plastic affair supplied with the DOS version of 3D Studio (*Fig 2*), but even if it turns out to be unsuitable, it will be possible to buy other renderers. Autodesk showed me a ray tracer, being developed by a Scandinavian company, which even in Beta form looked promising

I have only just started to experiment with the package, but one thing that is clear from the start, is that it demands a workstation-class system. My 16Mb Compaq Pentium didn't even come close. The company recommends 32Mb of RAM, a Pentium Pro (or two — MAX supports multiprocessor configurations) and graphics acceleration.

The key to MAX's success will be its object-orientated architecture. All registered users will be entitled to the Software Development Kit so, if literate in C++, they will be able to produce their own extras. A few plug-ins (no longer called IPASes) are already available — for example a VRML converter — and loads more are promised.

Character building

One of the most eagerly awaited MAX plug-ins is Character Studio, which comprises two tools: Biped and Physique. Judging by a brief demonstration of these, it looks as if they will provide a powerful system for animating humanoid characters. Biped is particularly interesting as it allows you to animate a mannequin simply by establishing the position of its footprints. You string a series of prints out on any surface and watch Biped work out all the steps, including ones that traverse stairs or uneven terrain. Character Studio is likely to cost several hundred pounds when it is released later this year.

A simpler and cheaper alternative is an intriguing product from Fractal Design (www.fractal.com), the company that brought us the excellent Painter 4 bitmap editor (see *Hands On*, June). Called Poser, the package comprises meshes for human figures, a library of poses that these figures can strike and tools to manipulate both figures and poses.

The elements that make up the figures are hierarchically linked (in the words of the old song: "The thigh bone's connected to the knee bone, the knee bone's connected to the shin bone…"), and these linkages contain an element of what the pros call inverse kinematics. When you move the hand, for instance, the rest of the arm follows in a way that is roughly equivalent to the movement of a real arm.

Despite these aids, manipulation of the figures can still be tricky, and I was soon forcing my unfortunate mannequins into bone-breaking contortions. The figures themselves are quite simple: you can have male or female, in sizes ranging from baby to "super-hero", and there is a library of poses, such as "fugitive" and "thinker".

In *Fig 3* you can see the baby male figure. Its mincing pose is one of those supplied in the library, entitled "model stance". The baby has been rendered using Poser's own renderer. This allows simple textures to be applied to the models, such as clothing (though it always has to be skin tight). Note that the baby, like all the models, is anatomically incorrect in certain vital respects.

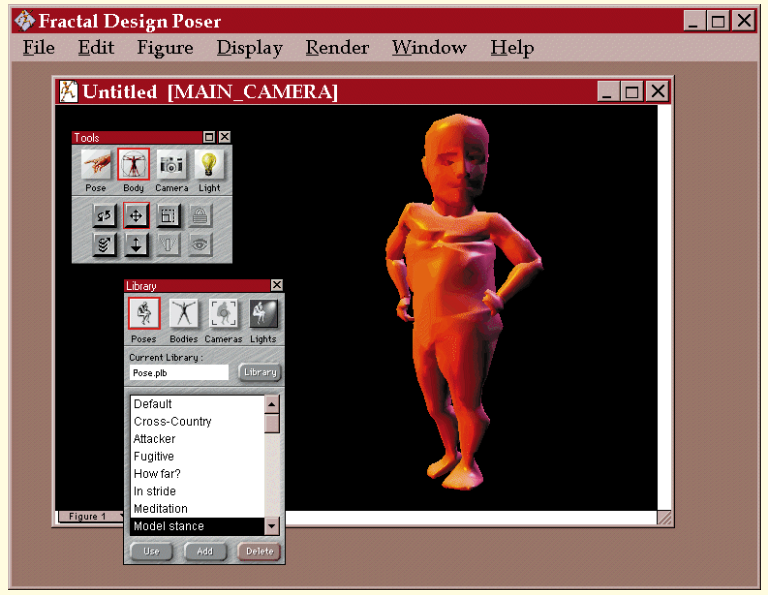


Fig 3 *Poser's interface, showing a rendered baby male figure (the difference from a baby female figure is not obvious)*

Poser's real potential for serious 3D users is for creating and posing models to be imported into other 3D packages. Unfortunately, the only 3D output formats are DXF and RIB (the Pixar Renderman format). DXF, essentially a CAD format, is common but crude. The file format does not contain mapping co-ordinates, which is a shame as all the Poser figures have mapping properly applied while used within the package (a texture map wraps around the figure no matter what pose it strikes). Poser is a nicely designed and well documented package, and if other file formats were included (the 3DS format, for example), it would be extremely useful.

Memory palace

In previous columns I have mentioned a project that I play with in my spare time entitled the "Memory Palace". My use of the term provoked the interest of a reader, Alan Mackay, a professor at the University of London. Memory Palaces were a medieval invention. They were not real buildings, but imaginary spaces created by story tellers in order to remember the sequence of events in a long tale. The tale would be broken up into rooms, each of which contained objects representing events in the story. By mentally retracing their steps through these rooms each time the story was told, the teller would recall the sequence of events.

The secret of a good memory palace is to make the rooms and the objects they contained as memorable as possible, which in medieval times meant filling them with violent and sexual images.

Nowadays, literacy is a little more widespread than in the days when memory palaces were created to help people retrace a long story. But as Mackay had noticed, the idea has applications to the way we store and retrieve information in our burgeoning disks and networks.

What if a system were represented not as a boring old office but as a palace made up of rooms containing striking icons and lurid objects representing the information the system can access? What if navigating the Web was not a matter of URLs and home pages, but a trip through an endless gallery of fabulous rooms, each furnished with objects acting as pointers to particular area of interest?

Several years ago, Mackay proposed a research project looking into some of these ideas, but the funding body for computer research at the time was not interested. Perhaps now is a good time to have another look at the issue.

PCW Contacts

Benjamin Woolley, writer and broadcaster, can be contacted at **woolley@illumin.co.uk**. His home page is **www.illumin.co.uk/woolley/**

No pain, **no gain**

Vibrant 3D images can cause file management headaches. Benjamin Woolley looks at ways of dulling the pain, and dips a toe in the water of the 3D interface with DIR3D.

This spring, Silicon Graphics announced its development environment for creating Web content, "Cosmo". Among the suite of fabulously sophisticated and glamourous tools (including tools for integrating VRML and Java) was a rather uninteresting-looking fellow called "MediaBase". All it apparently did was help you organise your files. What a dull job.

Well, unfortunately, that dull job turns out to be one of the most important in generating any sort of media-rich content and this applies, squared, to 3D. Think of it as the bureaucracy of beauty (if that is not too tortuous): to get those wonderful, colourful, incandescent, textured images, you will need a lot of files — and you will need to know where they all are and what to do with them.

I have yet to encounter a 3D graphics package, or even a utility for the PC, that takes all the pain out of file management (if anyone knows better, I would love to hear from them). But I would easily rate dear old DOS-based 3D Studio (3DS) as one of the worst.

When you install 3DS on your hard drive, it creates a series of directories for each of the constituents that are likely to make up a 3D project: meshes (the actual geometries for 3D models), materials (for materials libraries), lofts and shapes (both for building 3D models out of 2D shapes), fonts, images, maps (for texture maps, though sometimes these are to be found in the images sub-directory), processes (containing the IPAS routines, or "plug-ins" as they are better known in the rest of the graphics universe) — the list scrolls on forever.

Supposedly, saving a file as a "project" overcomes the problem of having to deal with all these different file types (materials, 2D shapes and other elements are stored in the one .PRJ file), but you still have to remember where all the texture maps are, the lofts and shapes you may have used in some earlier version of the project, and the clip models you may want to merge into the scene. Also, any large project is likely to comprise a number of smaller ones merged together.

Windows packages (such as Truespace, Visual Reality, Extreme 3D and Ray Dream Studio) overcome a few of these problems because you are better integrated, with a friendlier operating environment. You have the Registry and Explorer on hand to help. Also, with Windows 95 and NT, you can use long files names — an advantage you should exploit to the utmost.

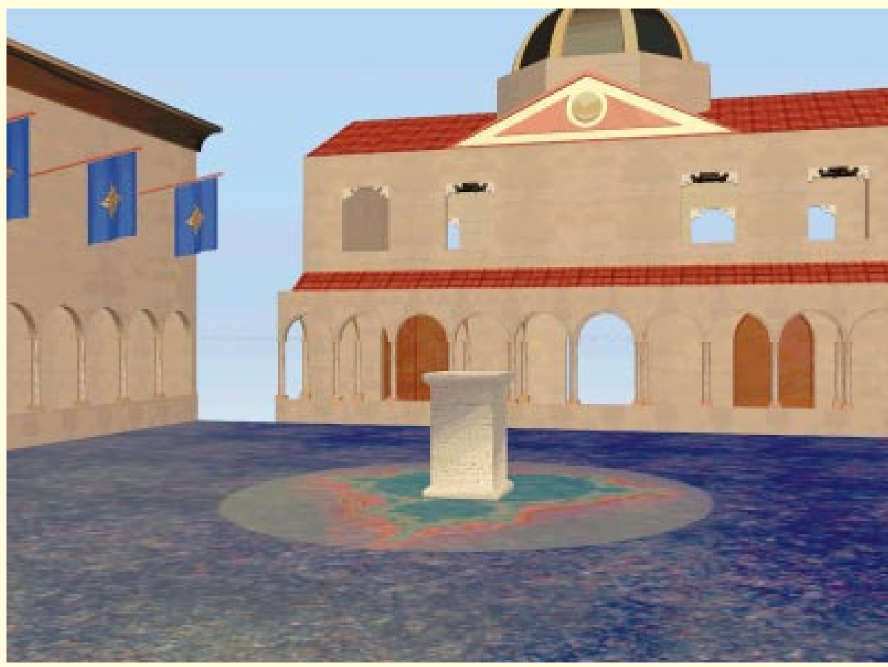
Avoiding problems

Nevertheless, no matter how disciplined you are, problems will still arise, so here are some ways of avoiding them.

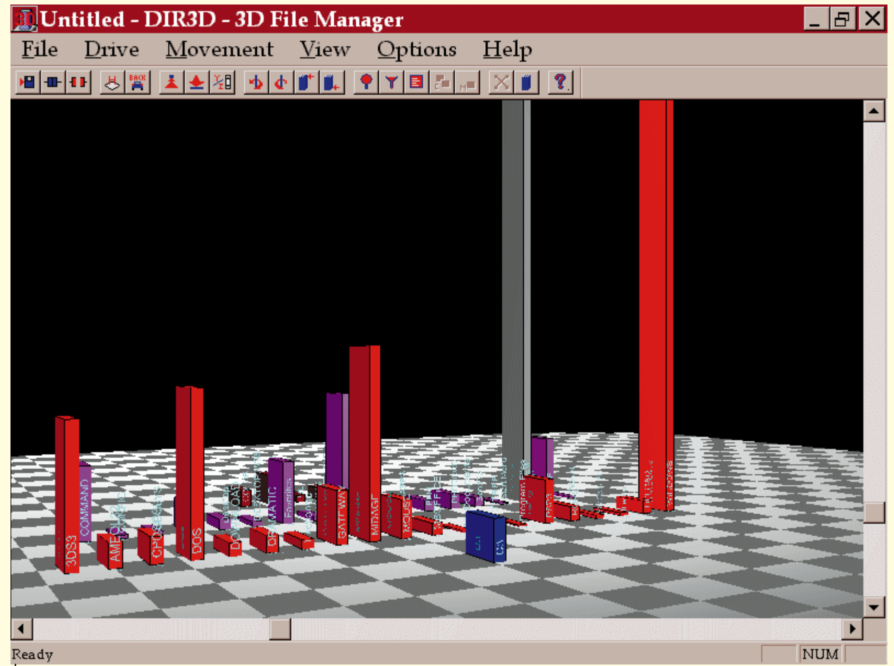
Firstly, there is the obvious trick of creating a single sub-directory for each project. However, it is often better to add some sort of structure to this directory, so texture maps are in a sub sub-directory called "maps", and so on.

Secondly, there is the less obvious trick of trying to work out in advance what types of files you will need to use. This depends on the sort of package you are using. Are there separate formats for 2D geometry, for example? Do texture files need to be converted into a particular proprietary format (as they do in Extreme 3D)? Can you make the conversions in advance and do you need to keep the originals? The answer to the latter is yes, if you cannot reconvert.

Thirdly, you need to think about texture and bump maps. These files, which will be bitmaps (in some cases, including video sequences and animations) are the ones that cause the most problems. The reason? They are often huge and you are likely to be getting them from all over the place: from a clipart library, off the Internet, from your own image directories, from scans and from other rendering projects.



It took a total of 23 separate files to make up this scene, including some project files for individual elements (e.g. the flags), more project files to render up textures (the stars on the flag material), images generated by other programs (the Mandelbrot used in the floor material, from a fractal generator), texture maps from clip libraries (the marble finishes), 3D mesh files containing objects used to cut out the arch shapes, and so on



The Manhattan skyline, as produced by DIR3D's view of my local hard disk drive. The two World Trade Center towers are, inevitably, my overloaded Windows and System sub-directories: see how they dwarf the 3D Studio sub-directory on the left of the picture. The toolbar contains navigation buttons

Worse, you will often accumulate several versions: a high-resolution colour version for the texture map, a 32-bit version for Alpha channel data, a low-resolution greyscale version for the bump map, maybe a traced version to form the basis of a 2D shape for lofting. You have to discipline yourself to performing a regular cull of these files, printing out (or writing down) scene details for each project so you know what you have used.

A good tip is to keep an offline backup (on tape, say) called something like "originals" where you lodge one good high-res copy of every image you use, when you first use it. Then you can afford to delete online files when you reckon you have no further use for them.

It is tempting to believe that such problems will not arise as long as you bung all the required files into your new project sub-directory and sort out the mess later (my usual strategy). If you have a spare gigabyte or two of disk space this might work, but in the real world you will soon find yourself having to make room for new materials the whole time, deleting and moving files on the fly, hoping you have kept all you need but never quite knowing whether that "bricktmp.bmp" was really just a temporary scratch file for the brick surface you used in an earlier version of the project, or the one you ended up using.

Unfortunately, being an anal-retentive is the only solution to the file organisation problem until some clever company produces a version of Cosmo MediaBase for the PC market. It is not a very glamorous product category at a time when everyone is wanting to be the next Netscape, so don't hold your breath.

Beyond the GUI

When Alan Kay and his cohorts at Xerox's Palo Alto Research Center came up with the design for the graphical user interface, it was but a short logical step from a two-dimensional space (a "desktop") into a three-dimensional space.

Researchers at PARC itself have toyed with this idea, producing proposals for what they called the "Information Visualizer… a user interface paradigm that goes beyond the desktop metaphor to exploit the emerging generation of graphical personal computers and to support the emerging application demand to retrieve, store, manipulate, and understand large amounts of information."

How, then, would one go "beyond the desktop metaphor"? You could have a 3D representation of an office with a 3D desktop, a 3D filing cabinet with 3D drawers full of 3D files, a 3D waste paper basket (wow!) and, down the corridor, doors leading into the 3D "offices" of other users in your network neighbourhood.

You could, borrowing from the metaphor used in the interfaces such as Apple's eworld online service, have a door leading out into a street with buildings representing different services: point and click at the library, and you are offered a series of information services; point at the bank and you get financial services, and so on. It sounds quite seductive but so far nothing much has come of the idea. Nevertheless, things may be about to change.

One modest first step into the realm of the 3D interface is DIR3D, a Beta version of which I downloaded from the Web site of the program's authors, Regnoc (www.regnoc.com). It's nothing more than a version of the Windows 95 Explorer or 3.1/NT File Manager, in which the contents of local and network drives are represented as a 3D bar chart with the height of the bars showing the size of the directory.

Regnoc prefers a more glamourous urban metaphor in its description of the program, calling each directory a "building", and each file within it known as "floors". Hierarchy (the relationship of directory to sub-directory, to sub sub-directory and so on) is represented by the z axis: the root directory is at the front of the scene, the next level of directories behind it, their sub-directories behind them.

You use DIR3D by moving around the city, finding the building (i.e. directory) and then the floor (i.e. file) you want. When you click on the floor, it slides out — the urban metaphor is beginning to collapse here. This selected floor can then be subjected to any of the usual file operations that you would use with Explorer: copy, move and delete. You can right-click on the floor to get the associated file's properties and run it (assuming the file type is registered).

As implemented in DIR3D, the 3D interface idea seems to be little advanced but the program demonstrates a couple of interesting things. First, it shows a potential use for OpenGL, the 3D renderer built in to Windows 95 and NT. OpenGL works efficiently on Pentium systems, enabling programs like DIR3D to create pretty solid-looking 3D environments on the fly (not Doom standards, but that will come). Secondly, DIR3D suggests some possible ways of using VRML.

As most people now know, Microsoft is planning to integrate Web browsing into Windows 95. It might be possible to integrate 3D browsing too, so the interface to your system could be a VRML scene populated with 3D shortcuts to local files as well as remote resources.

PCW Contacts

Benjamin Woolley, writer and broadcaster, can be contacted at **woolley@illumin.co.uk**. His home page is **www.illumin.co.uk/woolley/**

On the **face** of it

If your hardware won't allow you to create complex modelling, one way is to focus on textures in 2D. Paint packages can help, says Benjamin Woolley.

IT MAY SEEM A PERVERSE CLAIM, BUT one of the most important pieces of software any 3D computer graphics artist needs is a 2D graphics package, particularly on a PC rather than a workstation. This is because 3D graphics running on a PC will only work comfortably if you are using relatively simple models.

By relatively simple, I mean scenes comprising perhaps 10 or 20 reasonably complex models. On a 16Mb Pentium system, for example, a furnished room with a few chairs, a table, some pictures and the odd piece of bric-a-brac will start to cause some serious disk thrashing and sluggish response times.

The secret's in the textures

The answer to this problem is either to upgrade your machine to a workstation-class system (dream on), or to use simpler models more ingeniously (cheaper and more rewarding). The latter is done by concentrating your efforts less on models and more on the textures that you are going to drape over their surfaces. This is the reason for focusing on 2D graphics.

When an object is rendered, there are several features of its appearance that you can manipulate. For example, take a rectangular block, one of the simplest possible 3D objects that you can create. You can simply give it a colour, which will produce a rather dull picture of a coloured block. However, you can also apply a "texture map" to it, an image (or indeed, an animation or video sequence) which is painted over the image. This image can be applied once across the length of one side of the block, or "tiled" several times over its length. Thus you could put a photograph of someone on the face of the block to turn it into a rather chunky picture (or, if the image is an animation or video, the block could become a TV set). Or you could apply a picture of a brick and tile it over the rectangle to create a brick wall. All of this will have been done using the same, simple object.

You can get more ambitious and apply a bump map as well as a texture map. Bump maps use the level of light of each individual pixel in the map image to determine how much the surface of the object sticks out. For example, you could give the brick wall a rough surface by applying a speckly image to it as a bump map. The bright speckles will produce bumps, the dark specks, pits. The higher the contrast level of the bump map image, the rougher the surface will look when the model is rendered.

All the 3D packages I have encountered come with a supply of images to use as texture and bump maps, normally stored on CD-ROM. They are generally delivered in standard image file formats, so they can be edited using a standard paint program. It is exploiting this capability that is, I think, the key to successful 3D work on a PC.

Turning professional

There are several paint programs available, many of them good. You even get one free with Windows, which is perfectly adequate for simple tasks. However, for really successful work, it is worth investing in one of the more professional packages.

The CorelDraw suite is probably the best known. It is cheap and quite powerful. I have been a user for some years (drawn by the price), but find the software lacks polish. Even with version 4 I was still encountering bugs, so I never bothered to upgrade to 5 or 6. Perhaps they are better.

Two alternatives I have been trying recently are the latest versions of Adobe Photoshop (3.0.5, which runs very nicely under Windows 95) and Fractal Design Painter (version 4). *Fig 1* shows the texture and bump maps created for a curtain using Photoshop. The curtain is used in a structure being developed for my "Memory Palace" project (*Fig 2*).

Creating these textures took a lot of effort, but I think it was worthwhile. First, I needed a picture of an ornamental star. I

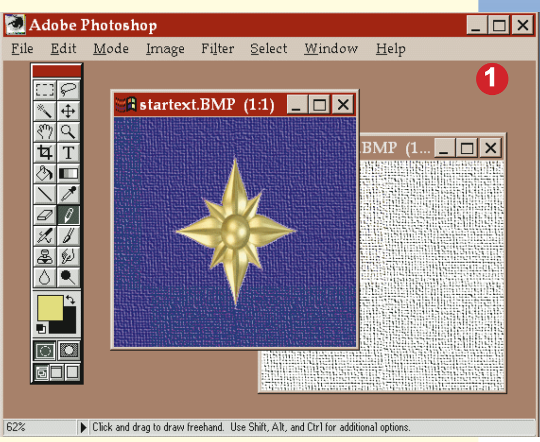


Fig 1 *Adobe Photoshop with star material and bump map*
Fig 2 *Rendered picture of the Memory Palace model, showing curtain*

3D packages: the art of seduction

People get very attached to their 3D packages. This is natural, as they have to spend so much time getting to know them. I apparently upset some veteran users of Newtek's Lightwave with my remarks about the package in the April column, so I have decided to try it out over the coming weeks. But I know already that after the first few hours I am likely to be seduced. Familiarity breeds devotion in this business.

I have found this to be particularly true with the 32-bit Windows packages Truespace (from Caligari) and Extreme 3D (from Macromedia). My loyalties have been flipping between these two ever since I started using them, and I still do not know which I favour (so, in true liberal fashion, I suppose I shall have to say I favour both).

Truespace has been around on the PC platform for longer (it is now in version 2) and its relative maturity shows. Extreme 3D, which is brand new, crashed a couple of times on my Compaq, though fortunately with frequent saving I managed not to lose anything vital. It also lacks support for all but one or two 3D and 2D file formats. Among other things, this effectively prevented me from using the Alpha channel in the creation of the curtain fabric (the Alpha channel is used to determine the transparency of each pixel in an image, so can be used to layer textures over the top of each other).

Such a serious omission is surprising in a package that is in other respects so well specified. Its texture-handling capabilities are extensive. There is network rendering (over mixed Mac and PC networks). It provides excellent realtime sketch rendering (just about the best in the business, I would say). It also offers full Bezier spline-based modelling, which is wonderful to use. The Bezier bit means you can edit curves with great precision and ease using "control handles".

Truespace also has splines, but it is in many other respects more basic than Extreme 3D, and this is reflected in the price. On the street it is well under £400 (at the time of going to press, Extreme 3D had a recommended retail price of £525). The documentation, in particular, is on the sparse side. However, simplicity has its merits. Truespace has most of what you need while remaining simple and straightforward. The most important tools are there, and all are a joy to use: intuitive, nicely organised, easy to access. Truespace also supports a vast array of file formats and makes a pretty good fist of translating them into its own.

Whichever is the better, both demonstrate at least one thing that relates to the first half of this month's column: the demands that 3D makes on the system. You cannot realistically run either package on a system with less than 16Mb. A Pentium is essential, as is a 24-bit graphics card and either a gigabyte of disk space or a removable disk (not for the programs, but for the vast array of files you will generate using them).

On my 16Mb 60MHz Pentium, state of the art just a year or so ago, the KISS principle applies whatever software I use: Keep It Simple, Stupid.

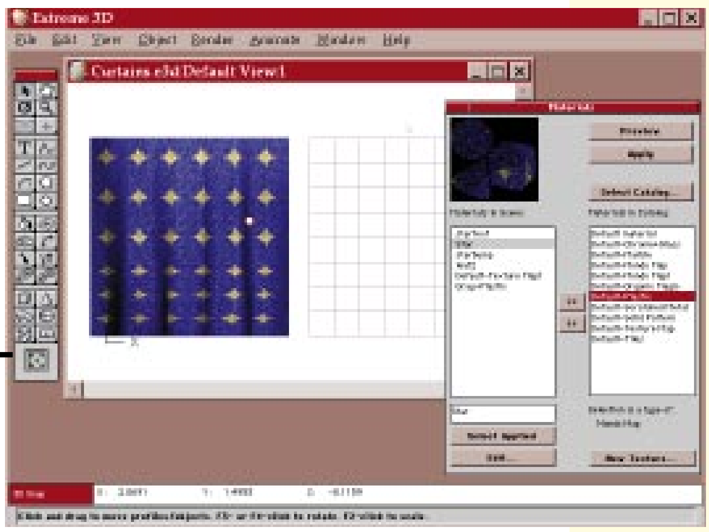


Fig 5 *Extreme 3D, showing curtain with material applied. The grid next to it is the working plane used to draw profiles for 3D models*

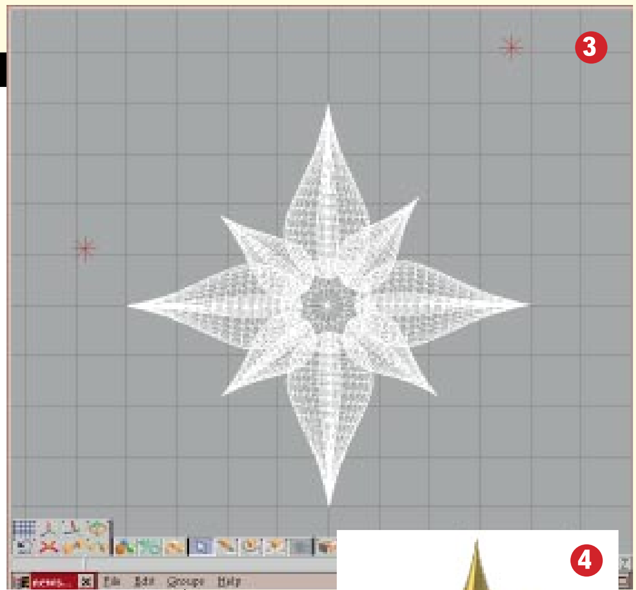


Fig 3 *Star model, displayed using Truespace 2 and ready for rendering*

Fig 4 *Rendered image of the star model*

decided to model one from scratch (*Fig 3*) and render the picture from that (*Fig 4*). I imported the picture into Photoshop and created a background of plain blue, which I gave a fabric texture using one of the plug-ins that are used to apply effects to images. This one was available from Adobe's new Gallery Effects range. I used the same Gallery Effect to create a fabric bump map (the underlapping window in *Fig 1*). This is in mono, because the size of bumps is determined by the brightness of each pixel rather than its colour (helps keep the size of the file down, which helps reduce rendering times later). You can see the preliminary results of using this material in *Fig 5*, applied and sketch-rendered using Extreme 3D (of which more, left).

Photoshop is excellent for editing and manipulating existing images. If you can't afford it, PaintShop Pro version 3, which is distributed as shareware, is a good substitute. To create images from scratch, however, it is also worth considering Fractal Design's new Painter 4, whch seems to be particularly good at creating various architectural effects such as mosaics and " tessellations" (good for stained glass). I'm still experimenting, but I have found it enormous fun. One criticism: some bright spark thought it would be entertaining to package the CD and manuals in a paint tin. This gives you a moment's amusement when you open it up, and hours of frustration as you try to fit everything back in again.

PCW Contacts

Benjamin Woolley - writer and broadcaster, can be contacted at **woolley@illumin.co.uk**. His home page is **www.illumin.co.uk/woolley/**.