4604ch03.qxd_sm 6/26/2000 10:58 AM Page 62

ENTERING THE THIRD DIMENSION

Topics in This Chapter

- An Introduction to the Human "Stereo" Visual System
- Using Monocular Depth Cues to Give the Illusion of Depth
- Projecting 3D Data onto 2D Surfaces
- Essential 3D Concepts and Jargon

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 63

Chapter 3

Although Web3D is relatively new, having only become practical for the everyday Internet user as of late 1998 (thanks to the maturation of enabling technologies described in Chapter 1, "Why Bother?"), the field of computerized 3D is decades old. And while 3D on the computer seems fairly modern, considering that it has only been in widespread use for the past dozen years or so, it is, in truth, simply one more step in the evolution of 3D that began thousands of years ago.

In this chapter I will describe how 3D "began" in the physical world when our primitive ancestors first focused their eyes in order to survive, found its way into the art world through eye-fooling techniques developed by Renaissance artists, and eventually entered the computer world courtesy of mathematics. In the sections that follow you will find that Web3D borrows heavily from 3D technologies that went before it, sharing many of the concepts, techniques, and terms common to the fields of art, mathematics, and traditional computer graphics.

This chapter sets the table for those that follow, giving you the implements that will let you consume the remainder of this book with ease. You will learn what the third dimension is, how it works, and why we need it. Along the way you will be introduced to the unique vocabulary of 3D, as we show you how the illusion of depth is projected onto flat, two-dimensional surfaces such as art canvases and computer screens, and how interactive 3D technologies allow you to *enter* the third dimension.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 64

Consumed by 3D

3D is a specialized form of computer graphics that has evolved over a number of years and can't be completely covered in a single book, let alone one chapter. The purpose of this chapter is to give you a basic understanding of 3D in preparation for the chapters that follow. Because computer 3D is fascinating, you may soon find yourself consumed by the concept. If you're interested in learning more about 3D in general, visit the following Web page. Here you'll find a number of links to print and online resources related to 3D:

http://www.web3dgallery.com/3d/resources.html

Wired for 3D

From the dawn of time human beings have been wired for 3D. As predators, we have two eyes spaced slightly apart on the front of our head, allowing us to drink in the same visual scene from two slightly different angles. Our brain receives these two visual images, one from each eye, and combines them into a single three-dimensional picture having depth; we "see" in 3D because of our stereo, or binocular, visual system (see Figure 3-1).

Through a relatively large binocular field of vision we're able to quickly perceive the physical, three-dimensional world around us. Our eyes and brain work in unison to allow us to see opportunity and threats instantly, which must have been a particularly handy skill in primitive times when it was necessary to hunt for food and hide from enemies as a matter of survival. Although we no longer struggle to stay alive on a daily basis the way our ancestors did, our 3D vision nonetheless remains essential in our modern, everyday life.

Just as it was for our ancestors, the real world we live in today is made up of three-dimensional objects (animals, people, trees, rocks, and so forth) located in three-dimensional space. This means that the things around us have depth as well as height and width, and they can be located close to or far away from us as well as to our left or right, or above or below us, or even behind us. It is the dimension of depth that accounts for what we commonly call the third dimension, an ever-present aspect of our world that we take for granted because it is an inherent and natural part of daily life.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 65

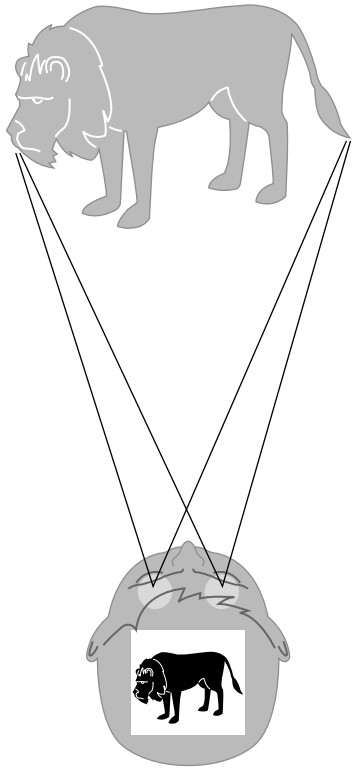


Figure 3-1 The human visual system is a stereo, or binocular, system that creates a single 3D image from two slightly different views of the same scene.

To reach out and pluck an apple off a tree, for example, requires a tremendous amount of 3D processing between your eyes and brain, even though you aren't conscious of what's going on. In order to successfully perform the task you must be able to judge exactly where the apple resides in 3D space and how large it is in all dimensions, in order to wrap your fingers around it. Furthermore, you must be able to instantaneously process any changes to the situation.

A breeze might come along and sway the apple to and fro, for instance, forcing you to dynamically recalculate an ever-changing location in 3D space. A strong wind can even blow your snack off the tree altogether, in which case you must track the apple's movement as it falls to the ground, where it bounces a few times and then rolls to a stop. Although it may seem trivial to follow the moving apple with your eyes, an astonishing amount of

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 66

Note: Eat or Be Eaten

The human visual system is similar to that of all predators. We have two eyes located at the top of our heads and slightly apart, giving each eye a slightly different view of the world. Our brain instantly and automatically resolves this retinal disparity *(also known as* parallax*), combining the images each eye delivers into a single three-dimensional picture, giving us binocular or* stereo vision*. Our binocular, predatory vision system is extremely well developed and is essential for hunting food that moves. Prey animals, on the other hand, have eyes located further apart, typically on each side of their head, which helps them "keep an eye out" for predators.*

3D processing is necessary to perform this simple, natural task. Imagine how much more your eyes and brain work when you trot across a street while avoiding oncoming traffic, or play a pickup game of basketball, or drive up to the takeout window of your favorite fast food joint for that matter.

Because we live in a three-dimensional world, our visual system automatically renders what we see in terms of 3D; we don't have to think about it, it just happens. We take 3D for granted nearly every waking moment of our lives. Representing the third dimension of depth on a flat, two-dimensional surface, such as an art canvas or computer screen, however, is a different story altogether.

3D on Canvas

The physical world around us is made of up three dimensions, but the same cannot be said for representations on the flat surface of an art canvas. Although our visual systems are highly developed and very sensitive to 3D information, artists did not commonly represent the third dimension on canvas until the Renaissance period. Whereas sculpting has always been a 3D art form since real-world materials used to create sculptures (typically clay) are inherently three-dimensional, applying paint to a flat canvas always produces two-dimensional art.

An artist cannot, for example, reach deep into a canvas and place a brush stroke of paint far away from the viewer, any more than he can apply paint in

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 67

Flat Surfaces and Monocular Depth Cues

In the real world our binocular vision is combined with single-eye, or monocular, depth cues to give us the full sense of the objects around us. However, on flat surfaces such as an art canvas, computer display, or photograph, only monocular depth cues are available to give us a sense of the third dimension.

mid-air to allow some brush strokes to appear closer to the viewer than others. Instead, every brush stroke is applied to a canvas on the same plane, where only the height and width vary. As a result, painting is by its very nature a two-dimensional art form.

Paintings created before the Renaissance are generally devoid of any sense of depth because of the physical limitations imposed by this two-dimensional medium. During the Renaissance period, however, artists such as Leonardo da Vinci developed and refined a number of visual techniques that could be used to give paintings the illusion of depth. Carefully examining the natural world around them, Renaissance artists found that they could trick the viewer's visual system by using a number of *monocular depth cues* found in our real world.

Monocular Depth Cues

Although we typically view the world around us using both eyes, which allows us to perceive the third dimension as described in the previous section, closing one eye does not result in a complete loss of depth. Try, for example, covering one eye while looking at a three-dimensional object such as the book you are now reading. Even though you will have lost the stereo vision that comes with two eyes, 3D objects do not suddenly appear flat and two dimensional when viewed with only one eye. However odd or unsettling it may seem at first to view the world with one eye, many people do in fact function quite well with only one, thanks to monocular depth cues.

Monocular depth cues are visual hints about the depth and location of objects that can be recognized with only one eye (hence the term *monocular* depth cues). Although they may have been used in the art world previously, the Renaissance period, and Leonardo da Vinci in particular, brought about

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 68

an unprecedented sense of realism in art by applying six specific monocular depth cues, which are used even today. Our visual systems pick up on these depth cues without any conscious effort on our part, just as our eyes and brain give us stereo vision automatically. Artists, however, both traditional and computer based, go to great lengths to imitate the following monocular depth cues using paints and pixels:

- **Size Differences:** Objects that are further away from us appear smaller than objects that are closer. This is why a car coming down the block seems to get larger as it approaches, even though it never really changes in size. Likewise, if you look down along a line of telephone poles, those that are furthest away from you will look the smallest, even though they are actually the same physical size as every other pole in the line. These size differences are visual clues that help us determine how close or far objects are from us, and where they are positioned in relation to one another.
- **Occlusion:** Objects that are closer to us can block, or occlude, objects that are further away. This seems to place objects "in front" of others, which tells us about their relative position to one another. A person standing in front of a building, for example, will occlude a portion of that building, which is a visual hint that tells us the front-to-back order of the scene. Because the person occludes the building, we can safely assume that he or she is in front of the building.
- **Lighting and Shading:** When light strikes an object that has depth, the surface of the object that is hit directly by the light appears to be the brightest, while the sides are shaded progressively darker. The back side of the object, meanwhile, is the darkest because it receives the least amount of light. The light seems to pour over the object to create gradients of bright to dark; light is most intense at the point of impact and gradually becomes less intense on surface areas further from the light source. If you shine a flashlight on a pumpkin, for example, the area where the light hits the pumpkin directly will appear to be the brightest shade of orange, while the sides of the pumpkin gradually darken and so are seen as progressively darker shades of orange. The back of the pumpkin, meanwhile, is the darkest shade of orange or perhaps even appears brown, because very little light hits that surface. Light sources also are often used to calculate shadows for objects, which can add a great deal to the sense of realism of a 3D scene. In cases where real-time shadow calculations are restrictive,

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 69

such as with interactive Web3D, 3D content developers can actually "create" fake shadows. (VRML and Java 3D, for example, don't support real-time shadow calculations, although experienced developers often create their own shadow effects in order to increase the realism of their content.)

- **Texture Density:** Real-world textures (such as the repeating tiles of a floor, shingles on a roof, or the stones in a wall) appear to become more dense the further away they are, which is a direct result of objects appearing smaller at greater distances (see Size Differences). If you are standing on a brick sidewalk, for example, the bricks beneath your feet form the texture of the walkway. The bricks directly under you appear to be larger and less densely packed than those that are further down the sidewalk. The brick texture seems to become even more densely packed with added distance, as the individual bricks that make up the texture appear smaller and closer together. Eventually the brick texture becomes so dense that you can't actually see the individual bricks, and instead see only their red color.
- **Linear Perspective:** Lines that are parallel to our line of sight, such as railroad tracks, appear to narrow as they recede, eventually converging at a point in the far distance known as the *vanishing point*. Although the Greeks discovered perspective almost a full millennium before the Renaissance, it was all but forgotten until artists such as Leonardo da Vinci and Albrecht Dürer used it to increase the realism of their artwork. The general concept of perspective in artwork actually combines linear perspective with size differences. Thus, in the case of railroad tracks, linear perspective is responsible for the tracks narrowing in the distance and eventually converging at a vanishing point on the horizon. Size differences, however, account for the railroad ties (the horizontal wooden beams between the tracks) appearing to become progressively shorter as they move further and further away from the viewer.
- **Atmospheric Perspective:** Objects that are close to us generally appear very sharp and detailed, while those in the distance are less detailed and often fuzzy. This difference in appearance is a result of light traveling through atmosphere; the further light travels, the more atmosphere it must pass through. Simply put, we "see" things as a result of our eyes receiving rays of light reflected by objects in our field of vision, which is the information our brain uses to form a picture of the physical world around us. Rays of light passing through

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 70

the atmosphere scatter when they collide with tiny dust particles and microscopic molecules of water and gas on their journey to our eyes, which can make objects look fuzzy and less detailed when compared to nearby objects that reflect light in a more direct path. In addition, light rays that travel long distances are more likely to also become refracted by heated masses of air, further distorting the information that reaches our eyes (mirages are the result of light rays that have been greatly distorted by heat masses). As a result, distant objects appear to be fuzzy while those close to us are sharper. Objects in the distance also tend to have a blue or monochromatic appearance, while the colors of nearby objects are typically more vibrant and vivid. This is a result of blue light rays having a shorter wavelength than other colors in the light spectrum, causing blue rays to be more easily scattered by atmospheric conditions than other colors (which in turn casts a blue tone over objects in the distance).

By applying these real-world monocular depth cues to their artwork, Renaissance artists elevated drawing and painting to an entirely new level. Realistic-looking artwork that appeared to have the dimension of depth was finally possible, thanks to these innovative eye-fooling techniques. Flat, two-dimensional surfaces such as art canvases and cathedral walls came to life through the illusion of 3D, paving the way for forms of computer 3D that would come centuries later.

3D on the Computer

Computer 3D is a relatively new phenomenon. Computers were born in research labs in the middle of the twentieth century and have only become "personal" and commonplace in the past decade or so. To put an even finer point on it, computer-based 3D has emerged for the general population only in the past few years, thanks to powerful, low-cost desktop computers combined with mass-market 3D technologies such as Web3D (see Chapter 1, "Why Bother?" for details). However, even though more than five hundred years have passed since Renaissance artists first learned how to use monocular depth cues to give their art a sense of depth and realism, the same basic techniques are used in modern-day computer 3D.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 71

Because the computer screen is a flat, two-dimensional surface, just as the art canvas is, the monocular depth-cue techniques that are effective in the art world are also effective in the computer world. The main difference, of course, is that paint is not physically applied to a computer screen. Instead, light illuminates picture elements, or pixels, of the computer screen, giving us what amounts to a digital canvas lit from behind.

Although the art canvas and our computer screens are similar in a few very important ways (both are flat, 2D surfaces used to display graphical representations of real or imaginary objects), they are more different than they are the same. The most significant difference lies in the fact that, at its heart, computer 3D is all about mathematics and number crunching. Three-dimensional computer content involves sophisticated mathematical manipulation of 3D data structures that represent objects and scenes having depth, which are ultimately projected onto a display device in a visual form the viewer can understand. Traditional art, however, is produced using real-world materials (such as chalk or paint) applied to a canvas by the hand of a human artist.

And while it is true that 3D computer content is also created by human artists using input devices similar to those employed by real-world artists (such as a mouse or stylus, which are the computer equivalent of a paint brush or pencil), the computer is an entirely different beast. Because the technology of computer 3D is so different, artists working in this field must learn an entirely new vocabulary and set of skills in order to master it.

Computer 3D Concepts and Jargon

Although computer-based 3D has become commonplace only in the past few years, it has been part of our high-technology landscape for several decades. 3D computer technology was hard at work long before the World Wide Web was even invented, transforming industries such as architecture, design, medicine, manufacturing, theoretical mathematics, and entertainment, to name just a few.

As a result of the marriage between computers and 3D, human beings have been able—for the first time in history—to create, manipulate, and explore three-dimensional structures without actually using real-world 3D materials such as clay, metal, or wood. Before computer 3D, for example, architects had no way of modeling buildings other than to create physical structures using real-world materials. Such physical models are time consuming and extensive to produce, as well as inflexible and quite constraining.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 72

Virtual Reality

Computerized 3D can be said to be "nothing more than an illusion," since the results are commonly displayed on flat, two-dimensional displays and rely on monocular depth cues in order to trick our visual system into seeing a third dimension. Virtual Reality (VR), however, is somewhat different.

True VR is typically experienced with the aid of 3D eye goggles, where each eyepiece displays a slightly different version of the same image (each image makes use of monocular depth cues, as do all other forms of computer 3D). The viewer's brain will automatically combine these two images into a single stereoscopic 3D image, which is generally more realistic looking than 3D displayed on a traditional computer screen.

Although this VR approach to 3D is more like our own biological stereo vision system, it's still an illusion; the objects and scenes created for each eyepiece in a VR system are no more "real" than those created on a desktop computer (the viewer's eyes and brain receive computer-generated images in both cases). However, stereo vision is actually achieved as a result of each eye receiving a slightly different version of the same scene, which the brain then resolves into a single 3D image.

Altering the physical model of a building in response to a client's feedback, for instance, takes a good deal of time and money, whereas a computerized 3D model can be altered with relative ease and at less cost.

Computer 3D also allows us to rapidly visualize concepts that might otherwise be impractical or impossible to construct in the real world. Accurately visualizing the molecular makeup of a new drug and how it might interact with the chemistry of the human body, for example, is practically impossible without the aid of computer 3D. Medical companies today routinely use computer 3D to design and develop drugs and medical devices at a fraction of the cost of traditional research and development, while surgeons can now simulate complex surgical procedures on their desktop computer before physically walking into the operating room. Similarly, mathematicians often rely on computer 3D to visualize complex mathematical equations that might otherwise be impossible to understand.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 73

Over the years computer 3D, like most computer technology, has evolved at a tremendous pace. Whereas the first 3D computer programs were merely the technological equivalent of simple drawing tools that could create simple 3D shapes, computer 3D technologies today are much more sophisticated and generally fall into three broadly defined categories:

- **Traditional 3D**, used to create 3D images and animation (sequences of images) that are viewed by the end user from a fixed perspective chosen by the artist.
- **Interactive 3D**, which allows the end user to navigate and explore three-dimensional scenes and the objects they contain from any viewpoint.
- **Immersive 3D**, which utilizes special input and output devices (such as video goggles and data gloves) to allow the end user to view and interact with 3D content in a way that closely resembles the real world.

Traditional 3D

Traditional 3D graphics programs create static computer art that has the illusion of depth using the monocular depth-cue techniques described earlier. These programs are the 3D analog to Adobe Photoshop, in that the images that they produce are just that: images. Unlike interactive and immersive 3D, which allow the end user to actually explore and manipulate three-dimensional objects positioned in 3D space, traditional computer 3D programs generate images that are viewed from a perspective chosen by the artist.

More sophisticated traditional 3D programs allow the artist to create animated image sequences, or movies, from a series of 3D images. However, just as with single static images the viewpoint is always predetermined by the artist creating the animation and can't be altered by the user. The best we can do is sit back and look at the image or animation produced by someone else; we're not able to interact with what we see.

However, despite a lack of interactivity, the content these programs create is usually of superior quality when compared to interactive and immersive forms of 3D. Because the viewpoint of traditional computer 3D is fixed (the angle of viewing does not change in response to user activity), the computer can spend a great deal of time drawing, or rendering, each image created by the artist long before it is displayed to the end user. Interactive and

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 74

immersive 3D are dynamically rendered in real time; traditional 3D is pre-rendered before the user sees it. A single image can take hours or even days to render as the computer crunches away at 3D data, yeoman's work that can produce wonderfully realistic results.

Traditional 3D is often used in television and motion pictures, as these forms of entertainment don't require input from the viewer; we just sit back and watch. However, the data crunching required to produce even one full minute of broadcast-quality 3D animation would bring a desktop computer to its knees. Because of the massive amount of processing power needed to create 3D such as this, television and movie studios typically rely on rendering farms—large networks of powerful, interconnected 3D rendering workstations—to generate 3D images and animation.

The 3D programs used to create broadcast-quality 3D are high-end and extremely expensive (prices start in the thousands of dollars and can quickly move into tens or even hundreds of thousands) and are often custom developed. You won't find programs such as these on the shelf of your local computer superstore. Artists literally spend years becoming proficient with these complex and very specialized 3D programs, and their skills are in high demand.

At the opposite end of the spectrum are mass-market traditional 3D programs, such as those used by Web page authors to create simple Web page buttons, images, and animations that have a three-dimensional appearance. Like their higher-end brethren, these products produce 3D content that is always experienced from the viewpoint selected by the author; you can't explore or manipulate such content any more than you can a GIF or JPEG image.

The advantage of these tools is that they are inexpensive (some are available as freeware or shareware, while others cost upwards of a few hundred

Note: Computer-Generated Imagery (CGI)

Traditional 3D computer graphics used in the production of broadcast-quality 3D images and animation are often referred to as Computer-Generated Imagery (CGI). George Lucas employed a fleet of 3D artists to create the Star Wars *prequel,* Episode I: The Phantom Menace, *of which approximately 90% of the scenes contained computer-generated characters, backgrounds, or special effects. Some feature films, however, are completely computer generated.* Toy Story, Antz, *and* A Bug's Life, *for example, were created entirely inside the world of the computer.*

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 75

dollars), easy to use, and don't require massive computing power to render the final results; a standard desktop computer is plenty of horsepower. The trade-off, of course, is that the images and animation produced by such low-end products are nowhere near as realistic as those created by high-end 3D graphics programs. However, they're more than sufficient when it comes to spicing up a Web page or exploring the 3D side of computer graphics.

Filling the giant gap between low- and high-end products are mid-level traditional 3D graphics programs that are typically employed to create 3D artwork used in print (magazine advertisements and product packaging, for example) and nonbroadcast applications. Mid-level packages are also commonly used to create introduction sequences for low-budget television programs and movies, for example, or materials distributed on CD-ROM, such as animated "cut scenes" that appear between the interactive segments of video games.

Although not as cheap as mass-market 3D products, mid-range 3D graphics programs are affordable when compared to high-end 3D products, typically ranging in price from a few hundred to a few thousand dollars. Mid-level 3D graphics programs are also more complicated to learn and use compared to low-end products, although they are not nearly as complex as high-end systems. Because mid-level 3D programs are more sophisticated and expensive than low-end systems, they're most popular with graphics professionals who don't require specialized products such as those used in the production of broadcast-quality 3D.

Despite the fact that they range in price, complexity, and capability, these forms of traditional 3D share a common purpose: they produce 3D images or animation sequences that are experienced from the viewpoint of the artist, not the user. Traditional 3D content has been fully rendered by the time you view it, meaning you can only look at it. You simply can't interact with traditional 3D.

Interactive 3D

As the name implies, the term "interactive 3D" describes a form of three-dimensional content that you can interact with. Unlike traditional computer 3D, interactive 3D is not prerendered or limited to a specific viewpoint. Instead, it is rendered dynamically ("on the fly") in response to user activity, giving the end user *control* over what viewpoint he or she sees the scene from.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 76

Panoramic 3D

In recent years panoramic 3D technology has emerged to straddle the ground between traditional and interactive 3D. Panoramic 3D consists of prerendered 3D artwork (or photographs) that depict a complete 360-degree scene. The individual images that make up the scene are joined together, or *stitched*, in such a way that the viewer appears to be positioned at the center of a circular scene or *image bubble.*

Imagine standing in the center of a hollow globe, the inside surface of which is covered with seamless images that together create a full 360-degree panoramic scene. Wherever you look you'll see a different portion of the scene.

Panoramic 3D give users the *feeling* that they are inside a 3D world that can be explored in any direction. Interaction with a panoramic 3D world is limited, however, as the images that make up the scene are prerendered and so prevent the user from moving about freely in all directions. Truly interactive 3D, on the other hand, is rendered on the fly as the user moves about, allowing for exploration in all directions as well as walking up to objects in a scene and examining them from any angle.

Interactive 3D comes in many forms. Web3D itself, as used in this book, is an umbrella term that covers several specific forms of interactive 3D, such as VRML, Java 3D, MPEG-4/BIFS, and X3D. Video games that use 3D technology are another form of interactive 3D, as are data-mining tools that display data in three dimensions instead of two. Likewise, many computerized Geographic Information Systems (GIS) take advantage of interactive 3D, as do many state-of-the-art network management tools, Computer Aided Design (CAD), and product manufacturing software packages.

The common theme that runs through these and all other forms of interactive 3D technologies is their ability to allow the end user to view a scene or visualize information from any perspective, not one predetermined by someone else. Interactive 3D also allows you to examine in detail those elements of a scene that you find interesting, just as you can walk up to objects in the real world and examine them from any angle.

Because interactive 3D is rendered on the fly in response to user activity, significant computing power is required to keep the display in sync with

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 77

Note: Restricting Interactive 3D

Authors of interactive 3D content can completely restrict the mobility and viewpoint of the end user if they desire, although they rarely do so, because severe restrictions defeat the purpose of interactive content. In general, content authors restrict user actions only when it makes sense in the context of a given scene.

A user might be restricted from walking in a certain direction if there is nothing in that direction to see, or prevented from walking through walls, for example. Similarly, users are often prevented from interacting with objects that they have no business touching (a pit bull guard dog, for example, might be off limits, while a French poodle could be approachable).

changes that occur in a scene as the result of a constantly changing viewpoint. In this sense interactive 3D is similar to our own visual system; in both cases what we see is computed real-time. Because computers powering most of today's interactive 3D pale in comparison to the computational capabilities of the human brain, however, interactive 3D as experienced by the general public is far from realistic.

However, thanks to advances in computing (see Chapter 1, "Why Bother?"), today's newly purchased home computers have the horsepower needed to deliver adequate interactive 3D. Using little more than a mouse or keyboard, users can interact with dynamically rendered 3D computer content delivered on fixed media, such as a floppy disk, CD-ROM, or DVD, or through networks. Web3D technologies, for example, typically deliver interactive 3D content via the Internet, yet can also deliver interactive 3D on fixed media (in the same way that standard HTML Web pages can be delivered over the Internet and on fixed media).

When experienced on a computer, interactive 3D content is generally displayed on a computer monitor and navigated with a mouse or keyboard. Console video game systems (such as the Sony Playstation) however, typically utilize a standard television screen for display purposes. Users of these systems don't require a mouse or keyboard to navigate and manipulate interactive 3D content, but instead use a specially designed controller device (similar to a joystick).

Although display and input devices used to experience various types of interactive 3D content may vary somewhat, in all cases the user remains in

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 78

Web3D Is Interactive 3D

Fundamentally, Web3D is a collection of 3D technologies designed specifically for distribution of interactive 3D content over the Internet. Although it's possible for various forms of Web3D to provide immersive 3D experiences (see "Immersive 3D"), the vast majority of people who experience Web3D do so using a standard desktop computer. In the future, however, immersive computer display devices may become commonplace, at which time Web3D technologies will be used to deliver immersive 3D content to the masses over the Internet.

visual contact with the real world throughout the experience. The average computer user, for example, typically sits about three feet away from the computer monitor, while console gamers often sit six or more feet away from the television display. Other forms of interactive 3D, such as information kiosks, also require the user to look at a video display situated some distance from the viewer's eyes.

In all cases, interactive 3D users are in constant visual contact with the real world around them through peripheral vision and eye or head movement (gazing away from the display, or turning their head). As a result, interactive 3D experiences don't give users the sense that they are actually *inside* the scene. Instead, they feel as if they are merely looking at computer-generated 3D imagery that they happen to control. They don't feel immersed in the experience.

Immersive 3D

Immersive 3D attempts to increase the realism of the 3D experience by walling off visual contact with the real world, essentially surrounding the viewer with computer-generated imagery. Both traditional 3D and interactive 3D are viewed within the context of the real world; people view these forms of 3D while also being able to see the world around them through peripheral vision or by moving their eyes or head. When you experience traditional 3D and interactive 3D, the experience doesn't envelop you.

Immersive 3D, on the other hand, heightens the experience by placing the viewer *inside* the display using special display technology such as video

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 79

goggles (see "Immersive Display Devices"). Video goggles, which typically resemble the rather bulky masks worn by scuba divers and downhill skiers, are fitted with a pair of diminutive CRT or LCD displays positioned just a few inches away from the viewer's eyes. Each eyepiece displays a slightly different version of the same 3D image (meaning each eye sees a single scene from somewhat different viewpoints), which are then processed by the viewer's own visual system in the same way real-world scenes are. The viewer's brain combines the two disparate images into a single, three-dimensional scene.

Because immersive display devices such as video goggles cut the viewer off from the real world, users generally feel as if they've truly stepped into the computer display. They can't simply turn their head or look sideways to escape the computer-generated imagery; they must either close their eyes completely or remove the immersive display that has taken over their entire field of vision.

As with traditional 3D and interactive 3D, there are a number of different forms of immersive 3D. The most simplistic forms merely present traditional 3D images and animation through immersive display devices, enhancing the realism of that content by removing the distraction of the world around the viewer. More sophisticated forms combine interactive 3D with immersive display devices and 3D input and feedback devices, and are often called Virtual Reality (VR).

VR is a vague, loosely defined term that is used to describe a wide range of synthetic, computer-generated environments. In one sense, any form of interactive 3D can be considered VR, simply because users can participate directly with such computer content (an alternate reality that exists only within the computer, making it virtual compared to our real world). However, to many in the field of computer technology, myself included, a true VR experience is much more than merely interactive 3D.

Purists believe that true VR is a convincing alternative to the reality of the physical world we live in, and, as a result, is the ultimate, most sophisticated form of immersive 3D. True VR tricks more than just our eyes; it manipulates many more of our senses in order to make the experience seem real, and as such it can be considered *fully immersive 3D*. Surround-sound systems, for example, can dramatically increase the realism of 3D environments, while specialized input devices such as head-position-tracking devices and data gloves allow our bodies to become part of the VR experience to create more fully immersive experiences.

Head-position-tracking devices allow immersive 3D displays to be updated in sync with the user's head movement (looking up, for example, will

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 80

automatically generate the appropriate display corresponding to such head movement). Data gloves, meanwhile, allow users to reach out and interact with the contents of a scene using their hands and fingers instead of a mouse. These types of 3D input devices can even be coupled with force-feedback output devices to further enhance the realism of a VR experience, allowing users to "feel" the objects that they handle in a virtual world.

Force-feedback devices can also stand on their own, acting only as output devices. The force-feedback vest, for example, is a wearable device that looks like a vest or perhaps a very thin life jacket. Most often used with immersive video games, it can produce the physical sensation of being struck on the torso, sides, or back in response to on-screen activity (such as a fist fight or gun play) without the unpleasant real-world side effect of internal bleeding.

Immersive 3D in general, and VR in particular, are the most specialized forms of 3D technology and therefore aren't yet commonplace. Because of the substantial computing power required, together with the high cost of immersive display devices (such as video goggles) and 3D input/output devices (such as data gloves and force-feedback vests), it will be years, perhaps even decades, before the average computer user partakes in any form of immersive 3D on a regular basis. Today, however, immersive 3D is making inroads in the entertainment world, particularly in high-end video-game arcades and theme-park rides, and also in professions that can justify the high costs and substantial computer-processing requirements.

The medical field, for example, has embraced immersive 3D (and even true VR) in an effort to reduce the risk associated with many complex surgical procedures. Surgeons can not only use immersive 3D to *practice* high-risk procedures (such as brain-tumor removal) in advance; today many can actually *conduct* such surgeries virtually through a computer rather than using their own hands.

Virtual surgery allows the surgeon to control various surgical implements (probes, scalpels, suction devices, and so forth) by manipulating 3D computer representations of these devices. Surgeons can guide these virtual devices through 3D data representations of the patient, such as a 3D tumor visualization created from Magnetic Resonance Imaging (MRI) scans of the patient's body. Traditional risk factors such as hand tremors of the surgeon can be eliminated altogether.

Today immersive 3D is generally limited to very high-end applications, such as specialty VR games available only at certain video-game arcades, expensive theme-park rides, engineering and simulation systems, and virtual surgery. It's only a matter of time, however, before the technology trickles

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 81

down to the consumer. Already, immersive 3D displays and input/output devices are available for home computer users, although they are quite expensive (a good pair of video goggles starts at about $1,000, for example) and there isn't much consumer demand for such products quite yet.

However, just as color computer displays and laser-writer printers came down in price over time while also improving in quality, eventually immersive 3D technology will come to the masses. I suspect that it will be several years, and more likely a full decade or more, before the average computer consumer steps into immersive 3D with any regularity. That day, though, will surely come—and, when it does, today's interactive Web3D technologies will have paved the way for immersive 3D distributed over the Internet.

Immersive Display Devices

Video goggles, the least expensive and most common type of immersive display device, are sometimes called "video glasses," "stereo goggles," or "shutter glasses." Video goggles are a modern-day embodiment of the "Ultimate Display" developed in the late 1960s by Ivan Sutherland. This was a Head-Mounted-Display (HMD) viewing device containing small video screens that allowed the wearer to become immersed in computer content.

HMDs are generally more bulky than video goggles and, as a result of hardware tracking devices typically built into them, tend to resemble battle helmets. When equipped with position-tracking devices, HMDs can report back to the computer any movement in the viewer's head position, for which the visual display can be automatically updated (thus, the HMD itself becomes a 3D *input* device that captures head movement). When the viewer's head turns to the left, for example, the HMD eyepieces are automatically updated in sync with that movement, similar to what happens when you look around in the real world. Although HMDs equipped with head-position-tracking devices can provide a much more realistic immersive 3D experience, they are highly specialized and quite expensive compared to video goggles.

The days of the Ultimate Display have long since passed, but today's video goggles and HMDs are direct descendants of Sutherland's novel invention. As we enter the year 2000, entirely new forms of immersive display devices have emerged, such as the 3D projection systems and optical lasers.

3D projection systems do away with wearable display devices such as video goggles and HMDs and simply project 3D scenes onto real-world surfaces, such as screens or walls. Today's most advanced theme-park rides, for

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 82

example, often use 3D projection systems to give participants the feeling that they're immersed in a fantasy world. As riders move along a closed track inside a theme-park building, various 3D scenes are projected onto screens or walls in sync with the moving car, giving riders the sense that they are part of the action as villains and heroes battle it out around them.

Similarly, CAVE Automatic Virtual Environments (or simply CAVE, which is a self-referential acronym) use 3D projection systems to provide unencumbered immersive experiences. A CAVE is a room whose walls are covered in rear-projected 3D scenes that are typically experienced in combination with video goggles and position-tracking devices. A CAVE can be occupied by more than one person at a time. Each person is free to move around the room, allowing for collaborative 3D environments and what's known as an unencumbered, or untethered, experience. However, typically the position of only one person in a CAVE is tracked, meaning each participant's view of the room isn't based on their individual viewpoint (unlike the real world).

A more radical approach to immersive 3D involves the use of optical lasers. Unlike display devices such as video goggles, HMDs, and projection systems, which deliver to the retina of the viewer's eye an image consisting essentially of reflected light rays, laser 3D systems use highly specialized laser beams to draw a 3D scene directly onto the viewer's retina. Such systems are very new and still in the experimental stages, yet hold great promise for the visually impaired. Because a laser excites the viewer's retina directly, such systems might one day provide computer-generated vision for the blind.

Note: Augmented and Enhanced Reality

Augmented, or enhanced, reality describes immersive 3D that is combined with real-world information. Virtual surgery, for example, often involves a combination of computer-generated 3D and real-world images. A 3D visualization of a brain tumor might be projected onto the real-world image of the patient's skull as he or she lies in the operating room. This would allow the surgeon to see a computer-generated 3D visualization superimposed on the patient's real-world body, making for an augmented, or enhanced, view of reality.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 83

Sound

Although most people think of "3D" only in terms of the depth (the third dimension missing from 2D graphics), sound also plays an extremely important part in the 3D experience, as it enhances the realism of interactive 3D and immersive 3D content. And while traditional 3D images aren't usually associated with sound, traditional 3D animation (a series of 3D images) is often enhanced with sound in the postproduction stage.

Interactive and immersive forms of 3D, however, don't have a postproduction stage; these forms are experienced live, in real time. As a result, sound can't be added to them after the rendering, as is often the case with traditional 3D animation. Instead, sound is typically an inherent part of the interactive and immersive experience.

Audio can be used in interactive and immersive 3D environments as simple *ambient* sounds that seem to have no particular source, or they may be very sophisticated forms of *spatialized* sound.

Ambient sounds are sometimes called *background sounds* because they aren't tied to a particular location or element of a 3D scene; instead, they seem to be coming from all directions. No matter where you are in a scene, ambient sounds have the same volume and direction. Blowing wind, for instance, is a good example of ambient sound because you can't really pinpoint where it comes from. Likewise, background music tracks (such as those used in video games) are typically played as ambient sound because there is no need for them to be emitted from a specific location.

Spatialized sounds, on the other hand, are tied to a specific location, or *emitter*, in a 3D scene and, as such, are more like the sounds we typically hear in the real world. The sound of a chirping bird, for example, is tied to the bird itself. As you move closer to the bird, the chirping becomes louder; as you move away, it lessens. If you walk around the bird, the chirping will differ, depending on your position relative to the bird. If you stand with the bird to your right, for example, your right ear will hear more of the chirping than your left ear. If you reposition yourself so that the bird is on your left-hand side, your left ear will then hear more of the chirping than your right ear will.

Spatialized sounds are emitted from a sound source positioned somewhere in the 3D scene, meaning that what you hear from your computer speakers (or headphones) will change depending on your proximity and position relative to the source of the sound. As you move around, the sound produced by your speakers is based on how far you are from the sound source, and also the angle

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 84

of your "virtual ears" to the sound source. Spatialized sound emitted from a trumpet in a 3D scene, for instance, is much louder when you stand directly in front of the trumpet than when you stand behind it.

Even though you may be the same distance from the trumpet in both cases, the volumes are different, because the sound waves emitted from the mouth of the trumpet hit your virtual ears directly when you stand in front. When you stand behind the trumpet, your virtual ears don't receive a direct blast of sound from the mouth of the trumpet, and so the tunes you hear aren't as loud. Similarly, spatialized sounds taper off, or *attenuate*, as you move further away from the source producing them. The further away you move from the trumpet, for example, the quieter it will sound.

Ambient and spatialized sound are used to enhance the realism of interactive and immersive 3D environments. While ambient sounds provide general background audio, spatialized sounds are often triggered by specific user activity and object behavior. Opening a wooden door to walk outside, for example, might trigger a creaking sound. Likewise, a car driving towards you might honk its horn if you are standing in the street when it is near, continuing to honk as it drives by until it's too far away to be heard. Meanwhile the ambient sound of rain might permeate the entire scene, filling the air with the soft music of raindrops to complement the visual treatment of a rainy evening.

Computer 3D Fundamentals

Although traditional, interactive, and immersive forms of 3D differ in their purpose and audience, at their core they are fundamentally the same: they're all forms of computer-generated three-dimensional graphics. Because all three stem from the same branch of computer science—3D computer graphics—they have many similarities beneath the surface.

Because the field of 3D computer graphics is the foundation on which various forms of computer 3D build, a number of concepts and techniques are common to traditional, interactive, and immersive 3D. The exact vocabulary used to describe these concepts and techniques may differ slightly from form to form and even from program to program, but the underlying fundamentals are surprisingly consistent.

Some 3D programs, for example, use the term "haze" to describe atmospheric perspective, while others use "fog" instead. These terms describe the same fundamental concept: computer-generated atmospheric perspective used to increase the realism of a 3D image or scene.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 85

In this section you'll learn about the fundamental concepts and techniques common to traditional, interactive, and immersive forms of 3D. In the process you'll find that the field of 3D computer graphics has a vocabulary of its own, one quite different from the 2D graphics world that you may be familiar with. Learning the unique and sometimes confusing terminology of 3D, however, is essential to understanding Web3D and the chapters that follow, and so we press on.

Modeling and Rendering

3D computer graphics can be broken into two primary steps: *modeling* and *rendering*. Modeling involves creating three-dimensional objects and arranging, or composing, those objects into a scene (see "Objects and Scenes" in the following section), while rendering is the process by which such content is actually displayed to the viewer.

Modeling can be thought of as the process of creating the architectural blueprints for a 3D scene, where each object in the scene is associated with its own blueprint that describes what the object looks like and where it is positioned in 3D space.

The act of modeling creates models, just as the act of sculpting creates sculptures. A *model* is a representation of some object or concept that captures its essential elements. A model train, for example, represents a real-world locomotive but isn't actually a working train itself. Instead, the model captures the essence of the real-world train by mimicking its appearance and perhaps even its behavior (how it acts).

Although models can be hand-coded in 3D programming languages (such as OpenGL, VRML, and Java 3D, to name a few), more often than not they are generated with visual modeling tools designed specifically for creating and manipulating 3D content (see Figure 3–2). Such modeling tools allow the nonprogrammer to create sophisticated 3D content, in much the same way as visual Web page development tools allow nonprogrammers to produce sophisticated Web pages without requiring detailed knowledge of HTML.

Unlike HTML files, however, there is no single, universally accepted 3D file format, just as there is no single, universal bitmap image file format. Over the years a number of different 3D file formats have emerged, and today most modeling tools support at least half a dozen or more to allow models to be easily imported, exported, and exchanged with other programs. Popular 3D file formats include AutoCAD Drawing Interchange File, Caligari trueSpace (COB), Digital Exchange Format (DXF), Lightwave Object

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 86

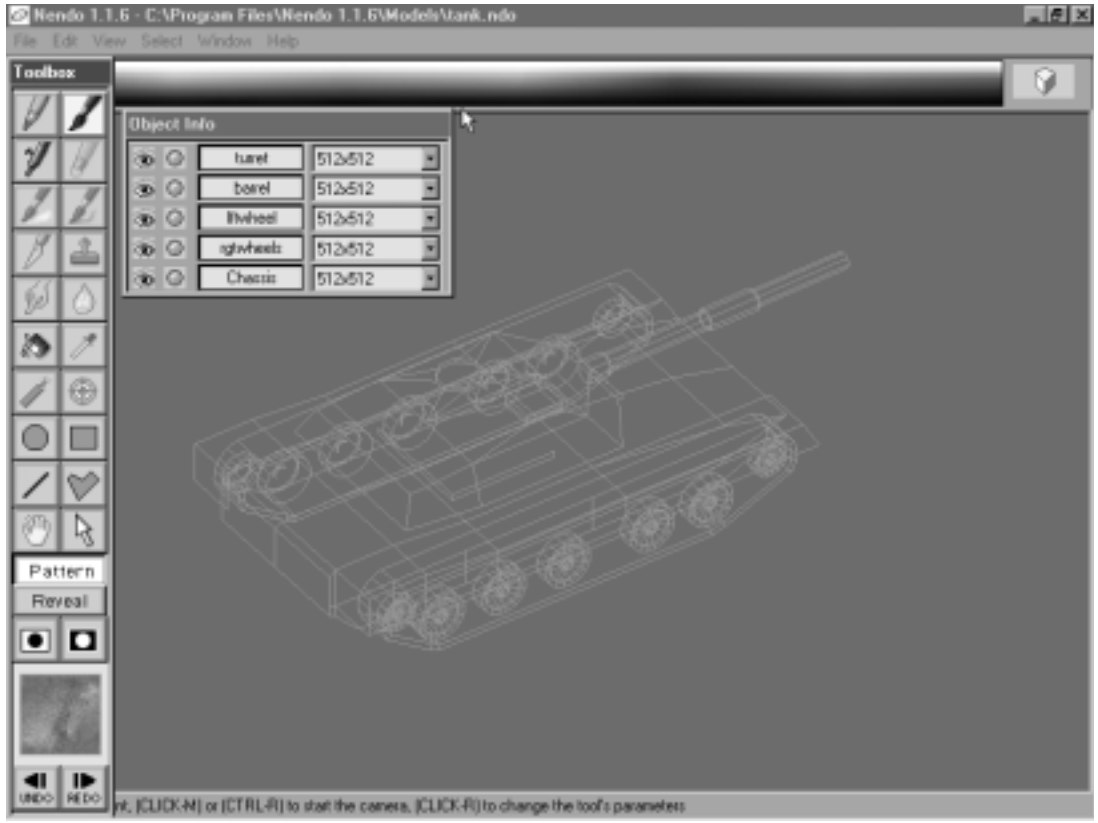


Figure 3-2 Visual 3D authoring tools such as Nendo can be used to create sophisticated models without the hassle of programming (Nendo image courtesy of Nichimen Graphics, http://www.nichimen.com).

Format (LOF), Lightwave Scene Format (LSF), Virtual Reality Modeling Language (VRML), Wavefront Object format (OBJ), 3D-Studio (3DS), and 3D Meta File (3DMF)—and there are many, many more.

Whereas modeling is generally considered the first step in creating 3D graphics, rendering is the last step; it involves the *rasterization* of 3D models. Rasterization is a rather fancy term that describes the simple concept of displaying an image on-screen based on the internal data the computer uses to represent that image (see Figure 3-3). Each pixel in a bitmap image (such as a GIF or JPEG image), for example, is stored in a computer file as a numeric value that describes a point location in 2D space (X, Y) and the color in which that point should be displayed. In order to display such a pixel to the

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 87

Geometric and Behavioral Modeling

Geometric modeling describes the process of creating objects based on geometry (shape or form). Behavioral modeling is used to define how objects behave (act or react). A geometric model of a car, for example, represents the shape and form of an automobile, while a car's behavior model defines what happens when you step inside and stomp on the gas pedal.

Geometric modeling can usually be broken down into six basic steps. First, the simplest parts of each object are created (in the case of a model car, for example, the simplest parts would be the wheels, bumpers, doors, hood, roof, windshield, and so forth). Next, these simple parts are composed into more complex objects (the car, in this example), after which surface and material properties such as color, texture, and reflectivity are specified (which control whether the car is red or blue, for example, and whether it is shiny or dull looking).

Objects are then positioned and oriented into a scene along with other objects (various car and urban objects might be arranged into parking-lot scene, for example), after which light sources are added. Finally, camera viewpoints are selected. After all of these steps have taken place, the model is ready to be rendered.

user, the computer must translate this numeric information into a point of light on the screen. The translation of image data into a visual representation that the user can actually see on the screen is known as the *rasterization process*.

The rasterization of 3D graphics is much more complex than that of bitmap images, however. The entire process, known as rendering, must take into account a great deal of information directly related to the third dimension of depth that 2D graphics lack. Whereas rendering a 2D bitmap image is a relatively straightforward process that involves translating numeric pixel values into on-screen points of light, 3D graphics rendering is considerably more complicated.

As the following sections describe, 3D graphics rendering involves the translation and projection of 3D model data onto a flat 2D surface plane. Like all computer data, 3D model data is stored in numeric form, although 3D models contain much more information than the simple numeric pixel

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 88

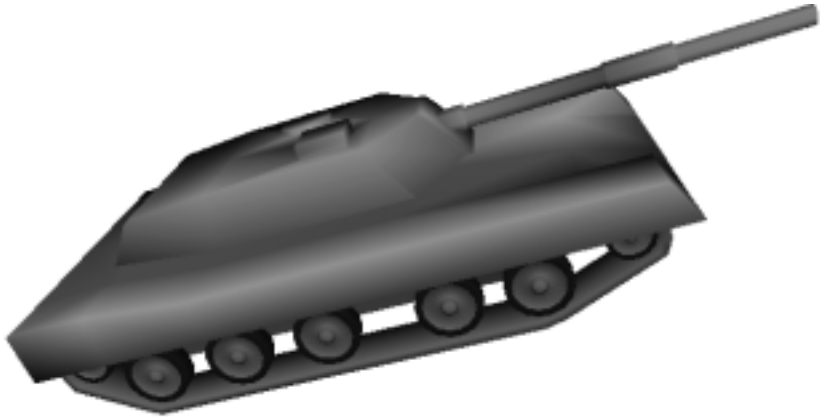


Figure 3-3 Rendering involves the rasterization of models, a process that ultimately draws pixels on the viewing device in order to visualize model data.

Raster Graphics

Raster graphics was invented by Xerox Palo Alto Research Center (Xerox PARC) in the early 1970s. Computer displays at the time were simplistic and similar in nature to pen plotters, as they were capable of tracing the outline of shapes with a beam of light (analogous to a plotter pen) but weren't proficient at filling these outlines in. Xerox PARC researchers devised a computer display that consisted of row after row of tiny phosphor spots that were lit from behind by a sweeping light beam.

These rows of phosphor spots were called *rasters,* a term borrowed from the mechanical press (the shelf holding a line of type in a mechanical press is a raster). The sweeping beam of light could turn phosphor spots on and off very quickly as it scanned each raster (row), one after another, allowing for shapes to be both outlined and filled. When the last raster was reached, the beam would repeat the process. Images created in this fashion became known as *raster graphics*, while the process of converting computer data into a light point on the screen is known as *rasterization* (also known as scan conversion or, more generally, rendering). Most computer screens used today are raster based, although the technology has progressed considerably since the early days at Xerox PARC.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 89

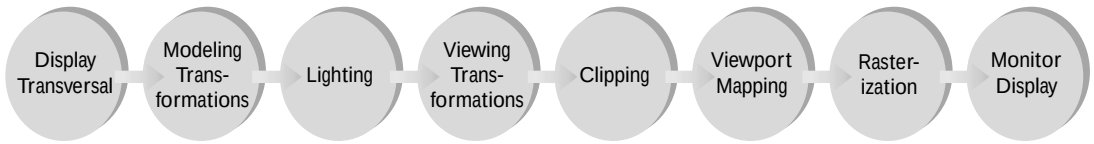
values used to represent 2D bitmap images. 3D models typically contain information about geometry (an object's form, or shape), location and orientation in 3D space, surface properties (such as color, texture, and reflectivity), and light sources, all of which are taken into account during the rendering process. In addition, interactive and immersive 3D models often describe the *behavior* of an object, which defines how the object acts and reacts in relation to the world and events around it.

All of these things, and more, can be contained in a 3D model, which the renderer must graphically display on screen. Such detail doesn't come for free, however, and requires much more processing on the computer's part during the rendering process when compared to 2D graphics. In order to digest this bounty of information, 3D rendering takes place in a series of steps or stages, much like an assembly line (see Figure 3-4).

The various steps that a model progresses through during the rendering process are collectively known as the *graphics pipeline* or *rendering pipeline*. At each stage the results are rendered into a *framebuffer*, which is an allotment of off-screen computer memory that stores the image as it develops. When the model has been fully rendered, meaning that the last stage in the pipeline has been completed, the contents of the off-screen framebuffer are transferred to on-screen memory so that the user can actually see the results.

Models can be rendered in a variety of ways, allowing for a great diversity in the visual appeal of a fully rendered scene. *Ray tracing* is among the most sophisticated of all rendering techniques, as this process literally calculates the precise path that rays of light follow as they bounce about a scene and interact with various objects. Extremely realistic results are produced with ray tracing, as light reflection, refraction, and even shadows are calculated during the rendering process, although the computational requirements are staggering. Ray tracing is typically restricted to traditional 3D, as light-ray calculations for extremely complicated scenes often require overnight processing by one or more computers.

Figure 3-4 The rendering process is actually a series of steps that a model progresses through before its internal data can be visualized on-screen.



4604ch03.qxd_sm 6/26/2000 10:58 AM Page 90

Other forms of rendering, such as *radiosity rendering*, can also produce high-quality results with equal or less computational effort. Radiosity rendering is quite useful in calculating the results of diffuse light bounced between reflective surfaces. The term *radiosity* refers to a light source that is reflected off objects in a 3D scene, making the light appear to be coming from all directions rather than a specific source, an effect sometimes referred to as *indirect* or *ambient* light.

If time permits, ray tracing and radiosity rendering might both be used, perhaps along with other rendering techniques (each being applied in a separate rendering "pass" that builds upon the results of the previous passes).

Because traditional 3D images and animation don't require real-time processing, they can be rendered over an extended period by more than one computer, or by so-called "rendering farms" as described earlier, and so can take advantage of sophisticated, photorealistic rendering techniques such as ray tracing and radiosity rendering. Interactive and immersive forms of 3D, however, are rendered in real time in response to user input and events (such as user movement and the passage of time) that continually alter the on-screen appearance of a scene; this places practical limitations on the rendering process.

Owing to their continual real-time rendering requirements, interactive and immersive forms of 3D use faster, less sophisticated rendering techniques that produce less visually realistic results than traditional 3D (light reflections and shadows, for example, are often omitted altogether or simply approximated). Calculated rendering-quality trade-offs are made in order to engage the user in a unique and compelling experience that traditional 3D simply can't deliver.

Flat rendering, for example, is a very fast technique that applies a single color to flat polygonal surfaces at the expense of realism. Other relatively fast rendering techniques, such as Gouraud shading and Phong shading, produce more realistic results. Gouraud rendering (introduced by Henri Gouraud in 1971) calculates the color intensity at the corner point, or *vertex*, of each polygon used to represent the surface of an object. These values are then used to average the shading applied across the entire polygon surface, meaning that pixel colors are interpolated based on vertex values, which results in a continuous, smooth transition of surface color. (Banding can be an issue, however, when tonal values change dramatically across polygon borders.) Although Gouraud rendering is a more realistic alternative than flat rendering, it takes more time and isn't perfect; it can't render texture maps (images) applied to an object surface, nor can it create shadows.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 91

Phong rendering was introduced by Phong Bui Tuong two years after Gouraud shading was first proposed. Like Gouraud rendering, it is an interpolated rendering technique, although values of reflected light are calculated for each pixel. As a result, Phong rendering can produce even

Rendering Engines

The component of a 3D-software program that is responsible for rendering three-dimensional model data is known as the *rendering engine*, or simply the *renderer*. Many 3D products use third-party rendering engines, rather than reinvent the rendering wheel themselves. Licensing a third-party renderer allows developers to focus their energy on other aspects of their program, such as the user interface or networking, rather than becoming bogged down with rendering details. Some rendering engines, in fact, are actually standalone software products (especially those used to create traditional 3D images and animation for film and video).

In all cases, from extremely sophisticated photorealistic techniques to simplistic real-time approaches, the rendering process takes into account three main aspects of a 3D model: geometry (shape or form), surface materials and properties (such as color and texture), and lighting (the effect that light sources have on a model). Generally speaking, the more time a renderer can spend on each aspect of a model, the more realistic the final scene will appear.

During the rendering process, the rendering engine must decide which parts of an object or scene are visible and which ones are "hidden" from view. A technique known as *hidden-surface removal* is used to determine what parts of an object are hidden, while *culling* describes the general process of eliminating entire objects from rendering. Imagine, for example, that you are looking at a 3D box shape head-on. In this case the back face of the box shape shouldn't actually be visible to you until you rotate the box or walk behind it. As a result, the renderer can hide this surface from you (a technique also known as *back-face culling*) until you need to see it. The box, however, may also obscure another shape entirely. If a cone sits behind the box, for instance, the cone can be culled entirely from the scene; only when you reposition the viewpoint in such a way that the cone becomes visible does the rendering engine actually need to display it on screen.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 92

smoother surface appearances as well as shiny surface highlights (commonly refered to as *specular reflection*). The Phong technique can also be used to calculate texture map effects on surface pixels, and, in some implementations, it can render shadows. Although more computationally expensive than Gouraud rendering, this technique isn't nearly as demanding as either ray tracing or radiosity rendering.

Objects and Scenes

3D modeling involves creating 3D objects and arranging them into a 3D space commonly known as a *scene* (3D space is also called a "world" or "universe"; see "Scene Management" later in this chapter for details). Once modeled, objects and scenes can then be rendered as described in the previous section.

The appearance and behavior of objects in a scene ultimately account for our final experience in viewing or interacting with a 3D work, just as the appearance and behavior of actors, actresses, and props determine the experience of a theatrical performance. In terms of their geometrical appearance, the Web3D objects that you'll encounter typically fall into one of three very general categories:

- **Primitive objects**, such as boxes, cones, cylinders, and spheres, are the fundamental building blocks of 3D computer graphics (see Figure 3-5). Primitive objects are the simplest 3D shapes that a 3D program can represent, as they cannot be reduced or decomposed into subsidiary shapes. Primitive objects are often used as the basis for more complex shapes. A human body, for example, can be represented quite easily with primitive objects (a sphere for the head, inverted cone for the thorax, a box for the pelvis, thin cylinders for arms and legs, and so forth). The resulting form, however, is far from realistic. As a result, primitives are often combined with machined and freeform objects to create more realistic looking shapes.
- **Complex objects** is a general term applied to objects that are more sophisticated than primitives. Complex objects come in many forms, such as machined and freeform objects. They can be created by combining primitive shapes together in a meaningful way, such as using spheres to construct a snowman object. They can also be constructed using other geometric shapes, such as points, lines, grids, polygons, and curves (see Figure 3-6). Often, complex objects are created using

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 93

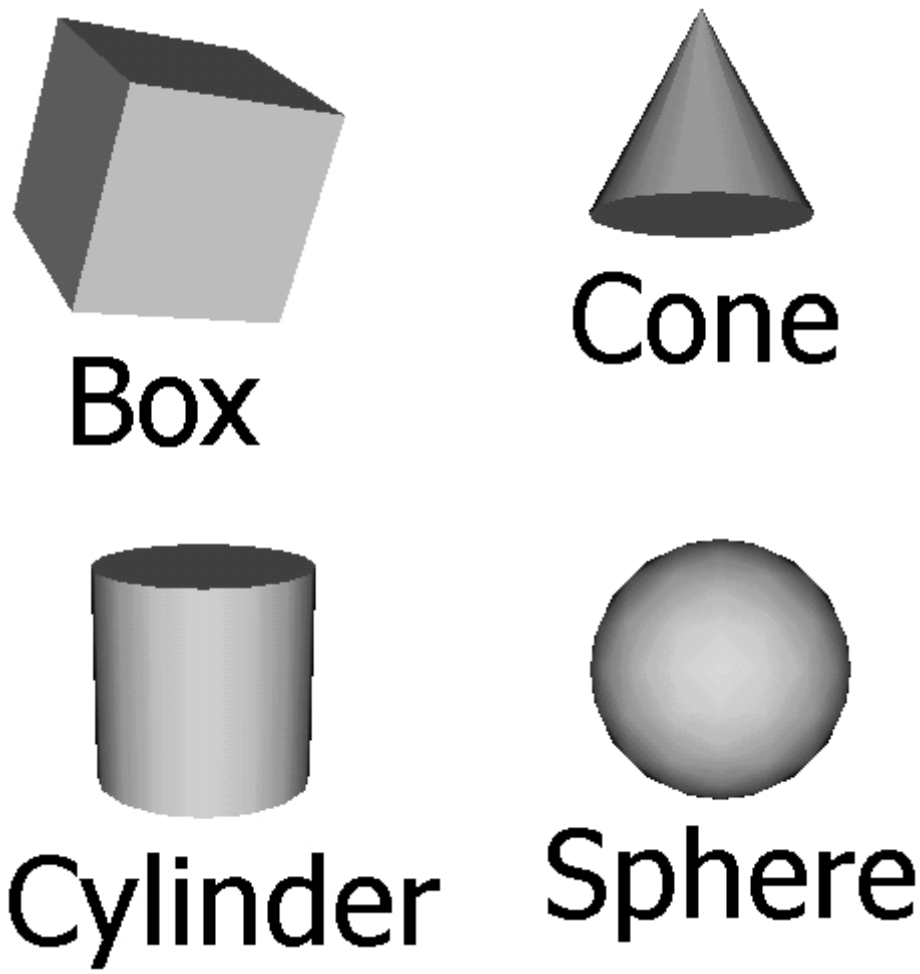


Figure 3-5 Primitives, such as the box, cone, cylinder, and sphere seen here, are the simplest predefined shapes that a 3D program understands.

a combination of primitives and geometrical shapes (the eyes of a human head object, for example, might be created using sphere primitives, while the rest of the head object could be defined by lines and polygons). Because the term *complex object* is so general, it is often used to describe any object other than a primitive. In this sense there are really only two types of objects, primitive and complex.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 94

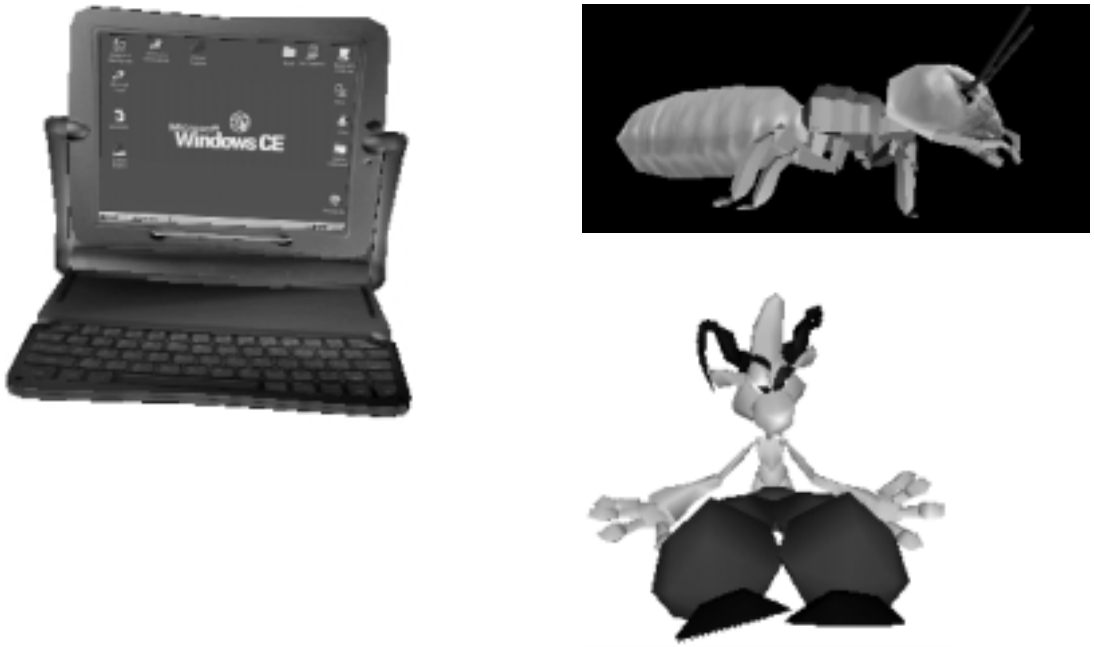


Figure 3-6 Complex objects such as these can't be represented using only primitive shapes and so are typically created using primitives, lines, faces, grids, and other 3D shapes (http://www.parallelgraphics.com/).

- **Machined objects**, such as extruded and lathed shapes, resemble items produced by these processes in the real world (see Figure 3-7). Noodles, moldings, baseball bats, desk lamps, vases, and pottery are all examples of machined objects created through an extrusion or lathe process. Although machined objects such as these are more complicated for computers to represent and render, they lend a sense of realism to 3D scenes that primitive objects simply cannot provide. Machined objects are created by *sweeping* a 2D shape known as a *cross section* or *profile* through space, a process similar in many ways to creating a bubble by sweeping a wand covered with soapy water through the air. Extruded shapes are created by sweeping the profile in a straight or curved path, a process much like that of squeezing cake frosting out of a tube that has a special nozzle attachment, such as a star shape (analogous to the profile or cross

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 95

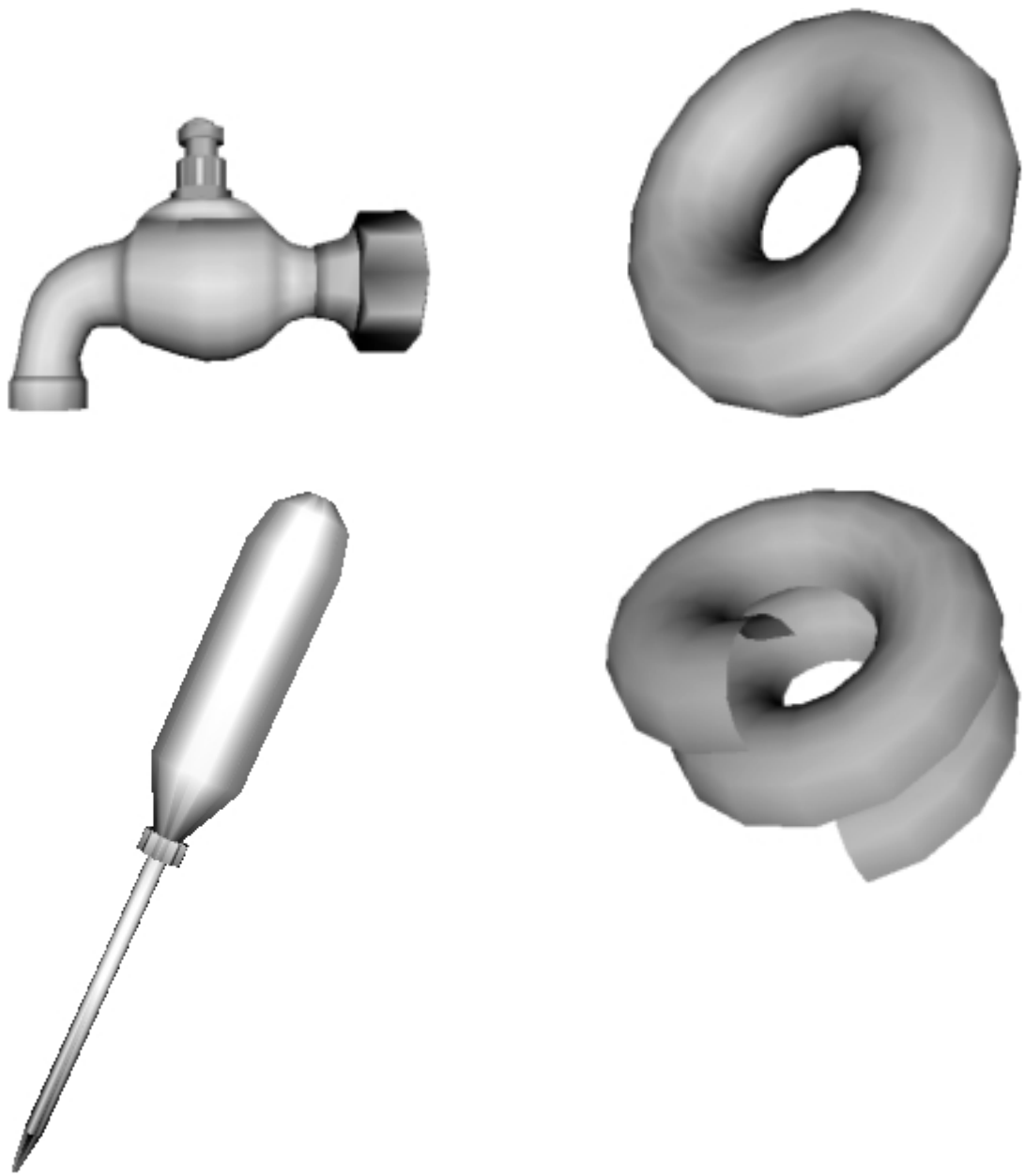


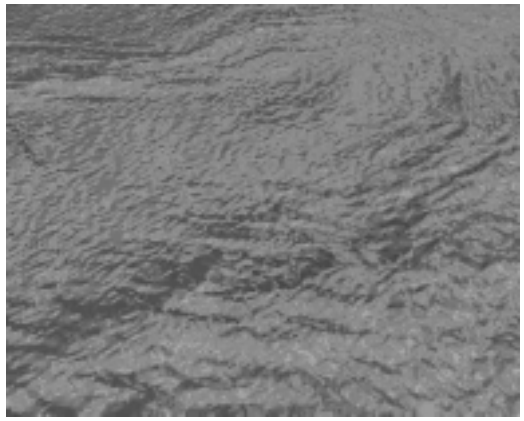
Figure 3-7 Machined shapes are typically created by a lathe or extrusion process, producing 3D objects that resemble those created in this way in the real world.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 96

section of a 3D extrusion). Lathed shapes, on the other hand, are the result of sweeping a profile along an object that revolves around an axis, much like a pottery wheel or real-world lathe. In both cases the 2D profile that defines the resulting shape can be dynamically resized and even reshaped during the process, allowing for a variety of appearances. A 3D baseball bat, for example, is a lathed object for which the circular profile changes size during the process, making the bat thicker in some areas and thinner in others while always maintaining a symmetrical appearance. A 3D spaghetti noodle, on the other hand, is an example of an extruded shape created by sweeping a small circular profile along a winding path. Because the circular profile never changes in size or shape, the resulting noodle is always the same diameter, although it twists and turns because of the path this profile sweeps.

- **Freeform objects** resemble shapes found in the natural world, such as water, fire, flower petals, terrain, landscapes, and other forms that are simply too complicated to represent using primitive or machined objects (see Figure 3-8). Many manmade objects, such as cloth, also fall into the freeform category because they are not easily constructed using an extrusion or lathe process. Freeform objects are generally complex, irregular shapes that require more computing effort to model and render than do primitive and machined objects.

Figure 3-8 Freeform objects often resemble shapes found in the natural world, such as water and terrain (http://www.web3dgallery.com/people/gq/).



4604ch03.qxd_sm 6/26/2000 10:58 AM Page 97

Freeform objects are typically produced as the result of mathematical algorithms or procedures (such as fractal generators, growth simulators, and particle systems), allowing for a programmatic, or procedural, approach to modeling and rendering.

Creating Objects

Objects can be created in a variety of ways. Some programmers hand-code their objects, while others use modeling tools that allow for point-and-click object creation. In other cases, models are created using special 3D digitization devices that actually sample real-world items and convert the results into 3D model data.

Points, Lines, Polygons, and Curves

Although 3D objects can be seen on-screen during the modeling process and upon rendering, at heart they are really nothing more than numbers that represent points, lines, polygons, and curves in 3D space (typically defined in terms of X, Y, and Z axes, as seen in Figure 3-9). Just as bitmap images are merely files full of numbers that must be translated into on-screen pixels (a process known as rasterization; see "Modeling and Rendering"), 3D object files are also filled with numeric information that must be translated and projected onto a display device in order to be visualized.

Objects are defined as points in 3D space, which are connected by lines to form polygons or curves. These points are known as *vertex points*, or *vertices*, which is a mathematical term used to describe an arbitrary point in space. Each point, or *vertex*, used to represent a 3D object is actually a three-

Note: Procedural Objects

Objects can also be described dynamically by functions, or procedures. Procedural objects, however, aren't as commonplace as those represented using points, lines, polygons, and curves. Because polygonal meshes are well known, and the rendering of such objects is relatively simple, this approach to 3D representation has become ubiquitous over the years.

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 98

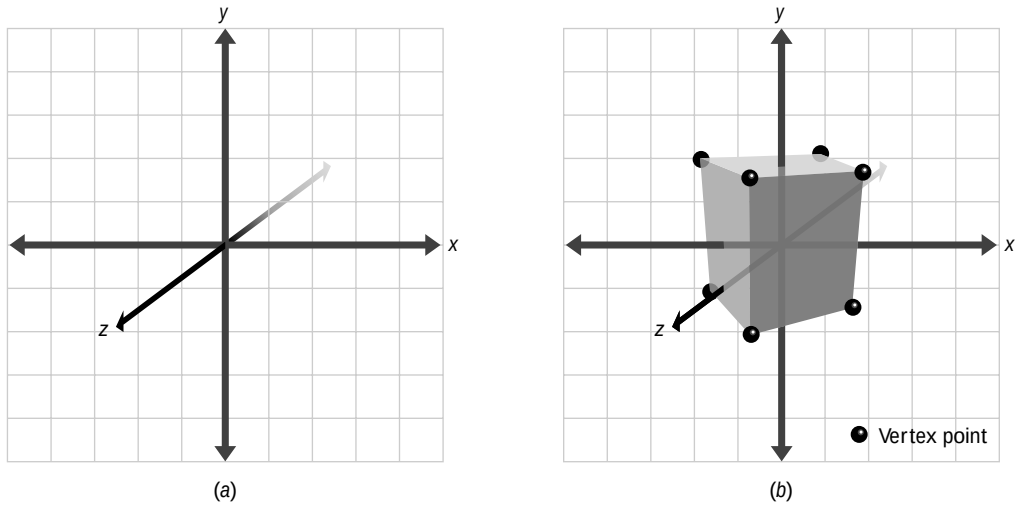


Figure 3-9 A vertex is a point in 3D space that corresponds to a given 3D coordinate (X, Y, and Z).

dimensional coordinate value that describes a location relative to three axes: X (width), Y (height), and Z (depth), as Figure 3-9 illustrates.

Vertices are the fundamental data values underlying all 3D objects and can be considered the low-level internal building blocks of 3D computer graphics. Although unimpressive individually, collections of vertex points can be used to define lines, curves, edges, facets, polygons, polyhedrons, and virtually all other structures used to represent 3D objects and scenes.

Two vertices (two unique coordinate values) can specify the beginning and end points of a line, for example, while multiple lines (defined by groups of vertices) can be joined together to form polygons. Multiple polygons, then, can be assembled together to form shapes, or objects (see the "Polygons" section that follows). Likewise, vertex points can be used to define curves, which can also be combined together into 3D objects (see "Curves"). Depending on the particular 3D technology in question, shapes are represented using either polygons or curves, or even a combination of the two.

Web3D objects are typically represented on-screen using polygons because of their relative simplicity and the fact that polygonal technology has been widely used in the 3D graphics industry for years (and, as a result, polygonal formulas and algorithms abound). Curve-based 3D-object representation, however, is somewhat more complex and has generally been reserved for traditional forms of 3D, although curve technology has recently

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 99

Vector Graphics

In graphics terminology the word *vector* refers to a line with a given direction, meaning that a vector is obtained from two points (a start point and an end point). The term *vector graphics* is sometimes used to describe graphics technologies that define images using mathematical definitions (such as PostScript image definitions that are interpreted by output devices such as laser printers), as opposed to *bitmap graphics* technologies that define images using a collection of raw pixel values (such as bitmap JPEG or GIF images).

3D technologies can be thought of as combining both vector and bitmap graphics. Fundamentally, 3D graphics involves the description of object geometry (shape or form) and appearance (color, texture, surface properties, and so forth). Interactive and immersive forms of 3D tend to take into account object behavior as well, which describes how an object acts. While certain aspects of objects' appearance can be represented in bitmap form (textures and color, for example), most characteristics of 3D objects are captured in mathematical formulas and descriptions that more closely resemble vector graphics.

begun to make inroads into interactive and immersive forms of 3D as well as Web3D. Behind both polygons and curves are humble vertex points that form the backbone of all 3D objects.

Polygons

Polygons are commonly used in the representation of on-screen objects because they are well known in the field of 3D graphics and relatively easy to render, whereas curves are more complex. Polygons are closed 2D shapes comprised of straight lines (which in turn are defined by vertex points) that do not intersect. 2D rectangles, triangles, and quadrilaterals, for example, are all polygons. Although a single polygon doesn't do much to represent a 3D object, a number of interconnected, variously sized polygons can quite literally paint a different picture altogether.

The surface of most 3D shapes can be represented as a *polygonal mesh*, which is a grid of connected polygons (a *polyhedron* is a closed polygonal mesh). A polygonal mesh is similar, in a sense, to a latticework of interwoven strings, such as a fish net. Each polygon in a mesh is known as a *face*. Faces give

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 100

Surface Properties

In addition to geometry, or form, objects typically have associated surface properties such as color, reflectivity, transparency, and texture maps (images that are applied to the surface of a shape). During the rendering process, these properties are combined with properties of the scene (such as light sources or atmospheric fog settings) to determine what an object will look like when displayed on the screen.

Although you'll hear us talk about object surfaces a great deal throughout this book, as these are what we actually see when Web3D content is rendered, some specialized forms of 3D are actually concerned with the *insides* of objects. Specifically, volumetric forms of 3D ("volume 3D") focus on the internal representation of objects as much as, if not more than, object surfaces. Volume 3D as applied to the medical field, for example, can tell doctors what's inside a 3D model of the human body, which is generally more important to a surgeon than the model surface. Web3D, in contrast, is concerned with the outer surface of shapes—the insides are generally considered to be hollow.

polygonal objects a faceted appearance when they are viewed without being fully rendered (imagine wrapping a fish net around an object), as seen in Figure 3-10. When an object is rendered, however, each face of the polygonal mesh is painted with the object's corresponding surface properties, which usually hides the individual polygons to reveal the overall shape of the object itself.

Some rendering techniques, however, don't completely hide the faces of polygonal mesh models. In particular, interactive and immersive forms of 3D typically rely on very fast rendering techniques that aren't able to hide an object's polygon mesh entirely. If you look closely at a VRML primitive sphere, for example, you will be able to make out the faces of the polygonal mesh used to represent these shapes. Likewise, many traditional 3D modeling programs rely on quick-and-dirty rendering techniques in order to allow the artist to create and manipulate objects in real time (reserving high-quality rendering for final production).

Wireframe rendering is often used during the modeling process, omitting color, lighting, and shading altogether in favor of raw speed. A wireframe view of an object is merely a series of connected lines, revealing every polygon in the mesh (see Figure 3-11). Because the surface properties of each

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 101

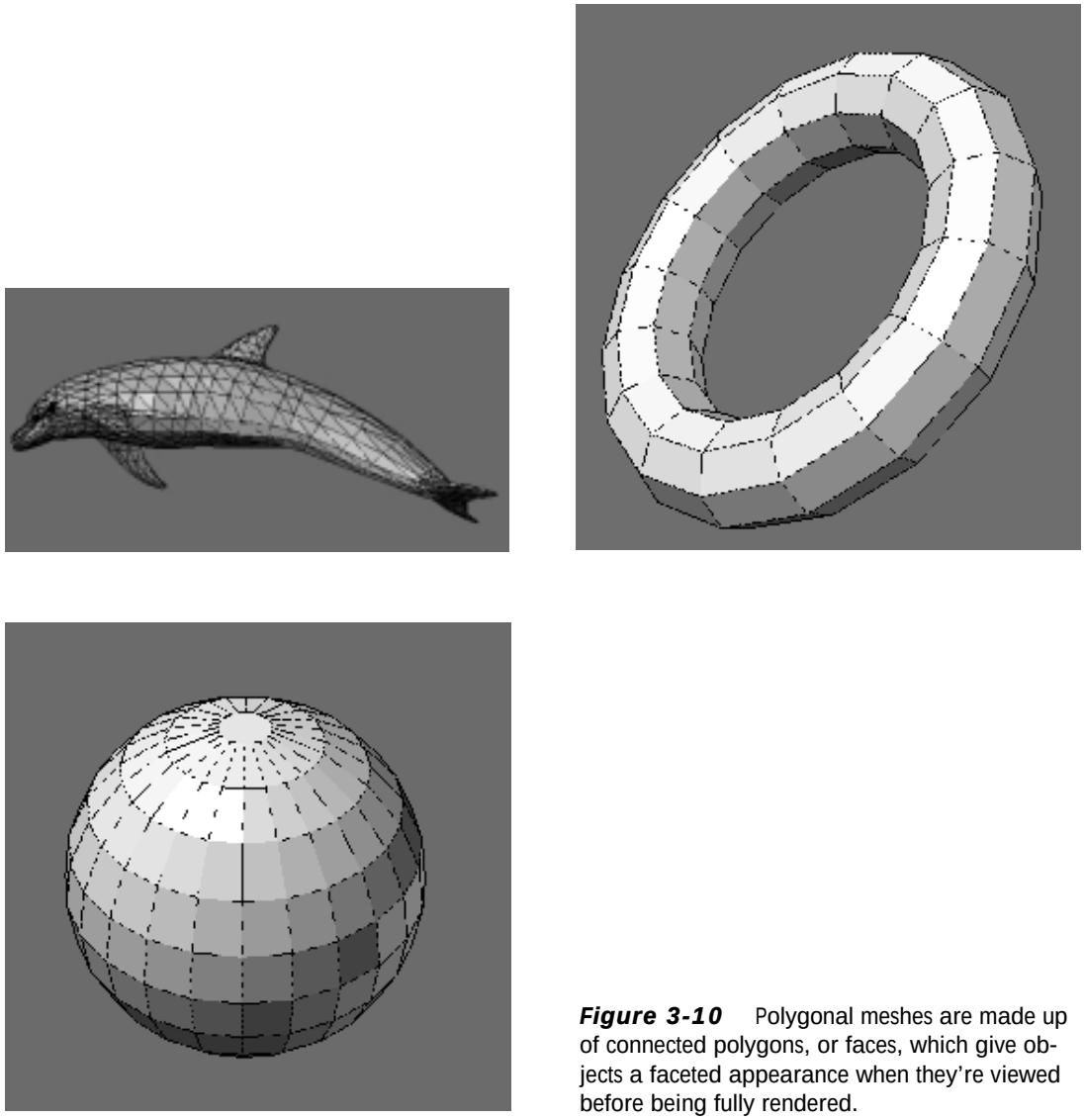


Figure 3-10 Polygonal meshes are made up of connected polygons, or faces, which give objects a faceted appearance when they're viewed before being fully rendered.

polygon in the mesh are not calculated and rendered on screen, wireframe rendering is exceptionally fast, and so it is useful in the design and development stages of modeling.

Although polygons are great at representing flat surfaces, such as the sides of a box or the walls of a building, they aren't so good when it comes to

4604ch03.qxd_sm 6/26/2000 10:58 AM Page 102

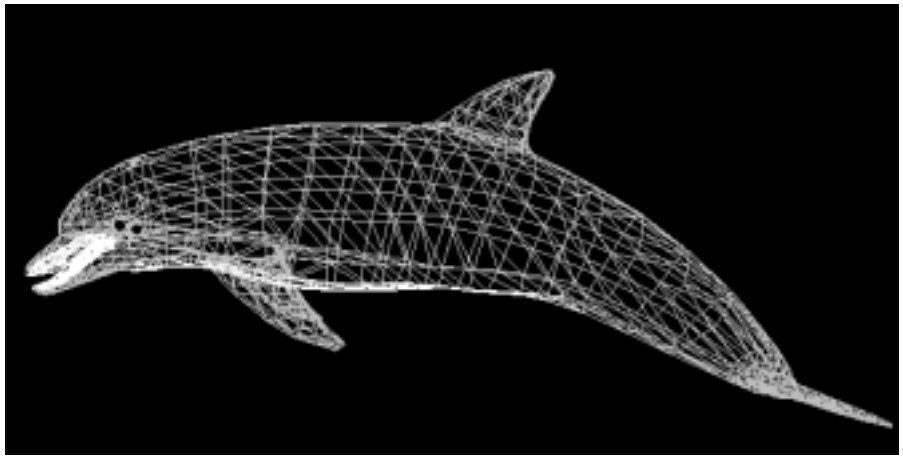
Note: Surface Patches

The individual polygons that comprise a polygonal mesh are sometimes referred to as patches, *or* surface patches. *These terms, however, are more commonly used to describe areas defined by parametric-curve algorithms (such as B-spline and NURBS algorithms).*

smooth, curved surfaces such as the natural curves and bends in the human body (e.g., elbows, shoulders, and knees). In order to create curved surfaces using polygons, the polygonal mesh faces must be quite small and plentiful; smaller polygons are used to create more finely detailed shapes. Unfortunately, a large number of very small polygons are consumed in the construction of smooth, curved shapes, which means extra work for the rendering engine. Because it is faster and easier to render a few big polygons than it is to render lots of tiny ones, objects and scenes rendered in real time often skimp on the number of polygons for performance reasons.

As a result, real-time polygon-based technologies (such as most forms of Web3D) tend to produce blocky objects when compared to non-real-time 3D technologies and those that support true curves. Traditional forms of 3D, for instance, are usually adept at creating smooth, organic shapes as a result

Figure 3-11 Wireframe rendering reveals every polygon face in a polygonal mesh.



4604ch03.qxd_sm 6/26/2000 10:58 AM Page 103

Polygon Budget

The number of polygons used to construct an individual object or entire scene is called the *polygon count*. Simple Web3D objects and scenes can be constructed with very low polygon counts (a primitive box, for example, has six sides and so requires only six polygons to represent it), although there is no restriction on polygon count. In your Web travels, you'll eventually come upon VRML objects constructed from hundreds and even thousands of polygons. These objects, however, require a great deal of computing resources to render, and can bring a modest desktop computer to its knees.

The number of polygons that a computer can comfortably handle at any given time is known as a *polygon budget*. Experienced 3D modelers are well aware of the polygon budget for the systems their work will be deployed on, and they do their best not to exceed this limit. Interactive and immersive forms of 3D targeted at standard desktop users have the most restrictive polygon budgets because of the real-time nature of these technologies (typical desktop computers and console gaming devices support on-screen polygon counts in the low tens of thousands and can even approach upwards of 50,000 on-screen polygons). Traditional forms of 3D, however, have very high polygon budgets because of non-real-time approaches to rendering (it's not uncommon for traditional 3D images to be constructed using hundreds of thousands, and often millions, of polygons).

It should be noted that polygon count is only one of many factors that affect rendering performance. Texture mapping, lighting and shading, and even sounds, for example, all contribute to overall system requirements for a given object or scene.

of their ability to handle a huge number of polygons and also because of their support for true curves. As desktop-computer technology becomes more powerful, however, real-time 3D technologies are seeing a corresponding rise in the number of polygons used to represent individual objects and entire scenes (see "Polygon Budget"). It's also interesting to note that curves have recently become a popular topic of discussion in the Web3D community, and it's very likely that the same form of curve technology will be directly supported by future forms of Web3D (such as X3D).

4604ch03.qxd_sm 6/26/2000 10:59 AM Page 104

Curves

Although polygons can be used to approximate curved surfaces, they tend to fall short of the mark when compared to the shapes created by a special class of curves used in computer graphics known as *splines*. Whereas polygons are a series of connected straight lines that do not intersect (such as triangles), splines are defined by points in space that control how a line is drawn.

The points used to define splines are commonly known as *control points*, which are used in the mathematical curve-generation process. Some types of splines are drawn as line segments that pass through each control point, while others come close to the control points but never actually pass through them. A specific form of spline known as the B-spline, for example, can pass through control points, or merely use them to influence the curvatures of line segments. Catmull-Rom splines, on the other hand, actually pass through all of the supplied control points .

B-splines of a particular form known as Non-Uniform Rational B-splines (NURBS) are often used in 3D graphics today. NURBS are constructed from control points for which extra parameter information may be supplied, and they are particularly useful in creating arced and elliptical shapes.

Scene Management

Although 3D objects can be interesting in their own right, they're generally destined to live in 3D space along with other objects. This space, when populated with objects, is commonly known as a scene, world, or universe. The main advantage of a 3D scene is that new viewpoints can be selected as needed, for which new renderings can be produced. As you walk through an interactive Web3D scene, for example, your perspective of the objects around you changes dynamically. Similarly, traditional 3D artists can position and reposition objects to their hearts' content, rendering anew each time. (Contrast this approach with 2D graphics, in which a change in object location or viewing angle typically involves substantial rework of the image on the artist's part.)

In order to accommodate an ever-changing viewpoint, 3D products must manage various aspects of the scene so that the rendering engine will know what must be drawn on-screen and what is hidden from view. Imagine, for example, that you're walking through an interactive 3D city. Such a scene might ultimately contain a number of skyscrapers, store fronts, apartment buildings, cars, taxi cabs, trains, and people. However, you don't view everything at once. Instead, as you move about the scene you'll see only those

4604ch03.qxd_sm 6/26/2000 10:59 AM Page 105

Object Representation

Although it may be tempting to think of objects in 3D scenes as being represented using either polygons *or* curves, in practice they're often used together. VRML, for example, does not support NURBS directly, although some VRML browsers (such as blaxxun Contact and ParallelGraphics Cortona) have extended the language to add support for objects created with these types of curves. Thus, VRML's traditional polygonal mesh objects can coexist with curved objects defined by NURBS. (Curves can also be used to describe the path that polygonal objects take when animated—a smooth animation path can be calculated using a mathematical curve formula, which the polygonal model might follow during the animation.)

However, polygonal meshes (also known as *polygonal nets*) and curves aren't the only ways in which 3D objects are represented; they're just two of the more common methods (with polygonal meshes being the most popular). A variety of other methods exist, including bicubic parametric patch nets that describe curved quadrilateral meshes and volumetric representation techniques such as Constructive Solid Geometry (CSG). For the purposes of this book, however, you need concern yourself only with polygonal meshes and, to some extent, curves.

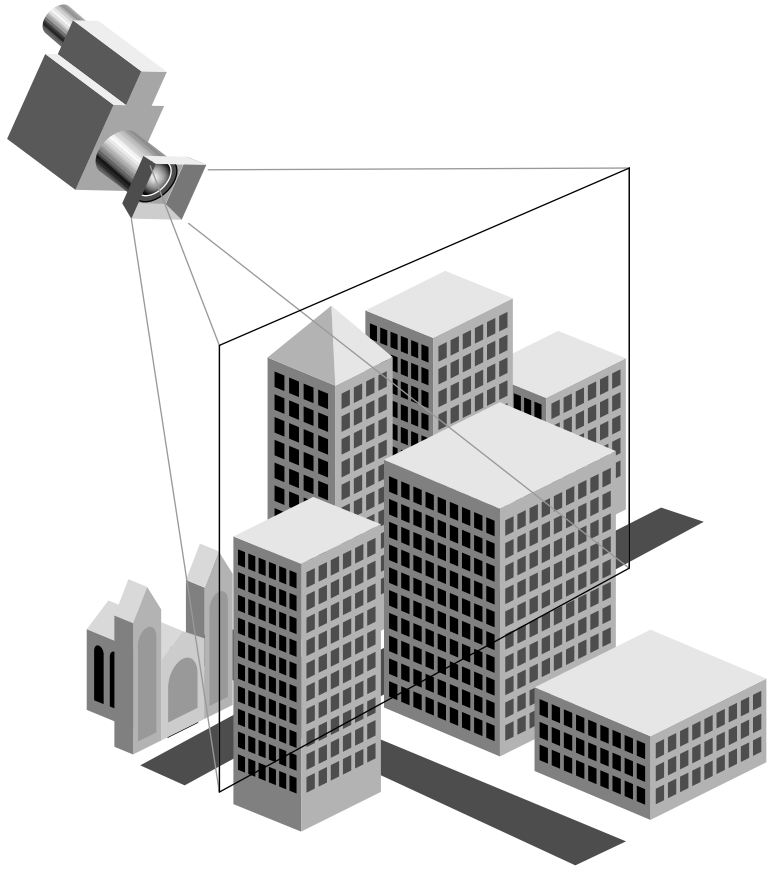
things that are in your field of vision. Those that aren't don't need to be drawn on-screen, but they must somehow be kept track of so that, when the time comes, they can appear.

Turning your head to the left, for example, will reveal an entirely different aspect of the scene than if you were to look to the right. Likewise, as you move forward in a scene, those objects that you walk past are no longer rendered on-screen, yet they must be available immediately if you were to turn around (at which point a completely different view of the objects would be rendered—the back view).

The concept of a virtual *camera* is typically used to construct a view of a 3D scene. Positioning a virtual camera in 3D space can be compared to pointing a real-world Polaroid camera at something in our physical world and then clicking the button to take a snapshot; whatever the camera's lens captures is rendered (see Figure 3-12). With interactive 3D, the camera is often continually moving in sync with the viewer (it can be said that the *camera* is

4604ch03.qxd_sm 6/26/2000 10:59 AM Page 106

Figure 3-12 Scene elements that are in the field of vision of a virtual camera are rendered to the display.



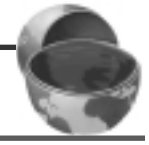
bound to the user's viewpoint); in such cases a better analogy might be a home movie camera, or camcorder. In both cases, objects captured by the camera lens appear larger as the distance between the camera and the object decreases, while those further away appear smaller. Similarly, the angle at which the camera is positioned relative to objects controls how the object will look. (Pointing the camera up at a skyscraper, for example, will result in a very different rendering than if you shoot the building head-on or from above, because the angle and position of the camera are different in each case.)

Although the notion of a camera can help us conceptualize what information gets projected onto our computer screens and what is omitted, it's also

4604ch03.qxd_sm 6/26/2000 10:59 AM Page 107

Note

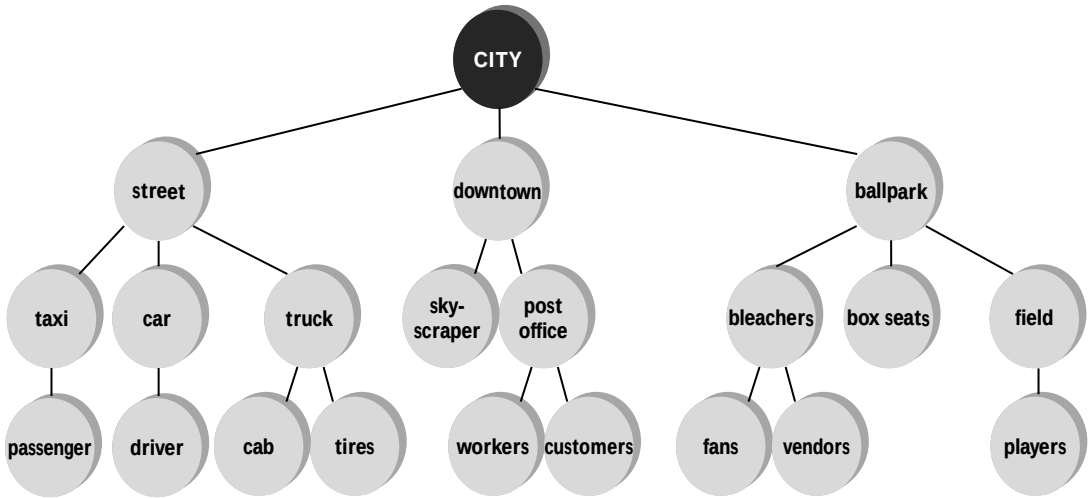
Cameras can be moved about a scene in order to create animation sequences. In such cases, the camera is detached from the user's viewpoint.



important to realize that there is some serious number crunching going on behind the scenes, both literally and figuratively. In order to manage all of the information inherent in a 3D scene, special data structures known as *scene graphs* are often employed (all of the forms of Web3D discussed in this book, in fact, take advantage of scene-graph technology to construct, store, and manage 3D scene information).

Scene graphs are hierarchical treelike data structures (see Figure 3-13) that describe an entire 3D scene and typically contain the geometric representation of objects as well as their appearance (such as color, surface properties, and textures). In addition, scene graphs commonly include related information, such as camera location and light sources. Because the information in a scene graph is organized spatially (meaning that objects are arranged in the data structure according to their actual position in the scene; their spatial relationship is reflected in the data structure), scene graphs allow common traversals to take place very quickly.

Figure 3-13 Scene graphs are hierarchical treelike data structures often used to organize and manage 3D scenes.



4604ch03.qxd_sm 6/26/2000 10:59 AM Page 108

A "draw" traversal, for example, can quickly determine what objects in the scene graph are actually visible on-screen, allowing this information to be sent to the rendering engine as needed. A "collision" traversal, meanwhile, can decide whether or not the viewer has collided with an object in the scene (such as a wall). These and other common traversals are very efficient, thanks to the way in which scene-graph data structures are organized, making such structures ideal for many 3D scene-management tasks.

Scene graphs also help shield programmers from the gory details of low-level 3D Application Programming Interfaces (APIs) such as OpenGL or Direct3D. Instead of focusing on complex 3D rendering issues, programmers can focus on *what* to render, not *how* to render it. In this respect, 3D programmers can "program to a scene graph" instead of a specific API, much as Java programmers can write cross-platform code without dealing with specific system-level APIs.

Summary

Computer 3D graphics is a specialized form of computer graphics that has evolved considerably over the years. Although human beings are wired for 3D, thanks to our highly evolved vision system, computer screens are flat, 2D surfaces. As a result, 3D data must be projected onto computer screens in a way that produces the illusion of depth. To create this illusion, computer 3D programs commonly employ monocular depth cues introduced and refined by Renaissance artists (such as size differences, occlusion, lighting and shading, texture density, linear perspective, and atmospheric perspective).

3D computer graphics can be broken into two primary steps: *modeling* and *rendering*. Modeling involves creating three-dimensional objects and arranging them into a scene (also commonly referred to as a world or universe), while rendering is the process by which such content is actually displayed on-screen to the viewer. 3D coordinate values (X, Y, Z), or vertices, can be used to define points, lines, polygons, and curves used in the construction of objects, although a variety of other object-construction techniques are also used.

Objects are typically represented internally by three-dimensional coordinate values that describe a location relative to three axes that together constitute 3D "space": X (width), Y (height), and Z (depth). Objects and their associated properties (such as colors, textures, reflectivity, transparency, and

4604ch03.qxd_sm 6/26/2000 10:59 AM Page 109

so forth) are often stored in a treelike hierarchical data structure known as a scene graph along with related scene information (e.g, light sources and camera positions). Scene graphs give programmers a convenient mechanism for managing 3D scene information and help to shield them from underlying system details.

Following is a complete list of URLs referred in this chapter:

Online 3D Resources http://www.web3dgallery.com/3d/resources.html

Nichimen Graphics http://www.nichimen.com

ParallelGraphics http://www.parallelgraphics.com

Gerardo Quintieri http://www.web3dgallery.com/people/gq/