3D Object Modeling

Norman I. Badler [*] Andrew S. Glassner [†]

1 What is 3D modeling?

Every computer-rendered image requires three essential components: a 3D scene description, one or more sources of light, and a description of the camera or eye viewing the scene. The scene description is typically composed of one or more *models*, or 3D structures. Typically we think of a *model* as a stand-alone part, e.g. a pencil or a tree, and the *scene* as the assembly of these parts into a complete 3D environment. This attitude reflects the most common procedure for building up a 3D scene: one builds many models, and then assembles them.

Each model contains two descriptions: a mathematical representation of the structure of the shape, and a recipe for figuring out how that shape would look if illuminated. Let us examine these descriptions in turn.

1.1 Model Structure

The structural description is basically one of *geometry*. It tells us where the object is in space, and where it is not. Imagine an empty coffee mug hovering in space in front of you. Now freeze time and look at every molecule in the room: generally each molecule will be part of the material of the mug or part of the air around the mug (note that the air molecules inside the mug where the coffee would go are not part of the mug itself; they're part of the air around it). If you paint every air molecule white and every mug molecule black, then you'll have a very fine description of the mug down to the precision of the molecules in the room.

The result of this thought experiment has many of the properties of 3D models used in computer graphics: it's conceptually straightforward (if bulky), and has a limited precision (most computer programs use the built-in calculation hardware in today's computers; this hardware has very high but limited precision). Some modeling methods are very close to this approach; they create points in space or chop up space very finely and label it empty or full. Other

[*] Computer and Information Science Department, University of Pennsylvania, Philadelphia, PA 19104 USA

[†] Microsoft Research, One Microsoft Way, Redmond, WA 98052

methods attempt to let you describe your model more abstractly, saying for example that your mug is basically a thick cylinder with a bottom and rounded lip.

1.2 Model Appearance

The other part of every model is the *surface description*. This boils down to describing the *physics* of how the surface of the model interacts with light. Happily, it turns out that these physics are driven by a few descriptive terms that have intuitive meaning for us, such as color, shininess, and transparency. Thus you could say that your coffee mug is shiny and blue, and go a long way towards describing its appearance.

Since models often simulate real world shapes, we have to decide how detailed we want to be. When describing the coffee mug's geometry, do we want to include every little bump in the ceramic and the irregularities in the overall shape, or is a mathematically perfect cylinder close enough? When describing its appearance, do we want to include the little scratches in the glaze, or is it enough to say that it is just uniformly shiny? The answers to these questions depend on how you intend to use the model. If it's going to be sitting on the shelf in the background of some scene, poorly lit and barely visible, then a blue cylinder might do the trick. But if the mug is the star of your image, front and center, large and well-lit, then you will probably want more detail.

2 Models: Images and Simulations

There are two principle reasons for building 3D models on the computer: image-making and simulation. These two procedures are very similar, but they typically have different goals.

Image-making requires models that *look good*. They need to be sufficiently complex that they aren't boring to the eye, and sufficiently realistic (when that's the goal) to pass convincingly as a good description of the real shape. The ultimate arbiter of the model's quality is the person who looks at the picture.

Simulation requires models that are *accurate*. Some simulations test two models mathematically to see if they fit together within some tolerance; the result is simply a yes or no answer, rather than an image. Consider a simulation that tests the strength of the roof of a new stadium under conditions of heavy rain: it's critical that the simulated roof have exactly the intended shape in order to determine how much rain will roll off the sides. An airplane wing also must be modeled with high accuracy if we expect to learn anything meaningful about its lift characteristics when actually actually built.

Some models are the result of a simulation: given a computer-controlled drill, a block of wood, and a program for the drill, what is the shape of the resulting part? Here we only supply the initial model, and the computer modifies the

model for us. We may be as interested in the surface finish of the final piece as its geometry. A hole might be round, for example, but we would like to be sure that it is smooth.

These two applications have spawned two rather different ways of describing models. One is interactive and approximate, appropriate for making models that just need to look good. The other is slower and precise, appropriate for models that must be accurate. Of course, there's a huge overlap between these different systems, since many models used for images need to be very precise, and sometimes simulations just need something close as a starting point or stand-in for a more complex shape.

3 Detail: How Much?

Details add interest. The amount of detail in a model can make the difference between a boring and fake image, or one that is interesting and realistic. For simulation, details can make the difference between getting an answer that is right or wrong.

3.1 Visual Detail

Early computer graphics images used very simple models: a table was a long box, made up of six perfectly flat, perfectly smooth, perfectly colored sides. Real tables are much more interesting because they have details. They have structural detail, and appearance detail (also called *texture*). Look at any man-made or natural 3D object, and you'll see a wealth of shape detail and appearance detail: nothing is perfectly flat, smooth, or featureless.

To make interesting and believable images, we need models that have enough richness that they begin to match the complexity of the real world (idealized forms are useful for art and education, but even there complexity can help you get your point across). The complexity of what seem to be simple forms can be staggering: look closely at the coiled line running from a telephone handset to the base. The wires follow a very complicated path, the gaps between coils increase and decrease, the coils themselves loop around and through each other. The finish on the wire isn't consistent either: there are scratches and discolorations all along its length, sometimes in random places and sometimes in organized chunks (like where the cord got caught and scraped in a doorway one day). It's this kind of detail that makes a model interesting, and creating this detail is one of the principle challenges to anyone who makes models.

In general, more detail in the geometry and the appearance is always better. But there are two caveats to this principle: more detail takes you (the *designer*) more time to create, and it takes the computer (the *renderer*) more time to process. In the previous example, a very detailed mug sitting in a dark corner will just slow down your picture-making and add nothing. The situations that

cause the most trouble occur in animations, when a shape starts out far away and barely visible, and later comes to occupy much of the image (such as a baseball hurtling towards the camera).

3.2 Simulation Detail

If you're building a model of a stapler and you want to make sure that the staples will be delivered correctly, it will be important to be very accurate in your model of the little hollow depression in the base where the staple folds. But if you only want to make sure that a pile of paper 30 pages thick will fit in the opening between the head and the base, the exact shape of the hollow won't matter. Finding the right level of detail is a matter of matching purpose to effort.

4 Modeling Software

There are several types of programs available to help you create 3D models. These programs are usually referred to as *modelers*. The person using the program is also sometimes referred to as the modeler, but that can be confusing (and dehumanizing). It seems more personal to think of the person as the *designer* (this is also more meaningful than the generic and uncomplementary *user*).

One idea that is common to most modelers is that the designer can create a copy of some basic form (such as a block or patch), and then modify it to create part of the model. The basic forms are called the *primitives* for that modeler, and each copy of is an *instance* of that primitive. So a blocky robot made of 20 rectangular blocks contains 20 instances of the primordial block. This approach is reminiscent of Plato's notion of ideal forms; the prototype shape is the ideal, and instances are practical realizations of that ideal.

4.1 Interactive Modelers

One class of modeler is *interactive*. Typically the designer sits in front of a terminal with a keyboard and mouse, and manipulates shapes on the screen. Because the image is only two-dimensional, different systems have different conventions to allow you to manipulate objects in 3D.

Interactive modelers typically contain a variety of design aids to help you achieve some precision in your model: a common example is to "snap" points on the model to other model points or to an invisible grid.

Models are built by interactively selecting, creating, modifying, and assembling instances of primitives. The shapes typically have "handles" that allow you to control the shape. Handles may come in the form of points, curves, or interactive tools.

Often one determines the surface characteristics when the shape is first created, but then textures may be laid on the surface to modify it.

4.2 Scripted Modelers

Another class of modeler is relies on an input *script* to define the model. This is typically a text file that has been created by the designer using any conventional text editor. The script specifies the shapes in the model one by one, identifying each one by its primitive, and the parameters that specify that instance.

Because scripted modelers allow you to specify operations numerically rather than interactively, they are ideal for very precise modeling. For example, it might be difficult to interactively place one block at a 37.5° angle to another, but it's trivial to specify when you can type that angle in.

4.3 Other Modelers

There are other ways to build models, for example by digitizing existing 3D structures, or by analyzing photographs. We will return to these techniques near the end of this paper.

5 How are models used?

There are many uses for 3D models. While there is an obvious desire to experience an object visually – hence computer graphics – there are other important motivations that form a larger and broader context for studying object models. Some of these uses (with examples) for 3D models are listed below. Note that each one tends to emphasize either the visual or simulation aspect of the model as discussed earlier.

- To visualize designs: How does a product look before it is manufactured? What is the best combination of shape, color, layout, etc.?
- To assess appearance: How does some existing environment look if changes are made to it? Does it need more or less lighting? Are the vistas appropriate?
- To observe part relationships: How do things fit together? What does an exploded view look like? Do parts touch or collide that should be separated?
- To check feasibility (of manufacture): Does the object meet the design specifications? Can it satisfy constraints on stress, loading, heat transfer, etc.? Can it be effectively machined, cast, or sculpted?

- To determine cost, volume, area, machining time, etc.: How much material is needed? How much material must be removed by machining? What is the cross-sectional shape of various internal structures in a complicated assembly?
- To determine faithfulness to physical phenomena: How does the surface of the object interact with light? How do the surface properties (smoothness, roughness, waves, etc.) depend on the geometry of the object? How close in appearance to "reality" can we get?
- To exercise display algorithms: What limits might we have on the computations over the object models? What determines graphical algorithm complexity? What display algorithms are needed to create images of a particular kind of model?
- To express artistic goals: What juxtaposition of the real and imaginary objects will protray an artists' visual message and mood? How much can the imagination be stretched to visualize artificial worlds?

6 The major issues

The major issues involved in object modeling include the computational cost of the model, its effectiveness in modeling the desired phenomena, its implementation complexity, and the methods used to acquire (or create) data that describe the object geometry.

The computational cost of a model may be gauged in terms of computer storage space, object construction time, display time, or in application use. For example, the storage space of a model may be based on the space required to store each primitive and the number of primitives needed to represent it. Thus a sphere is a relatively simple primitive; collections of spheres grow linearly with the number of spheres in the model. Polygon models may require variable length storage for an (arbitrary sided) polygon, plus additional link structures to organize a set of polygons into a closed surface.

In object construction time, the costs may based on where the data is and how the data is captured. Clearly more automated data capture can favor one object model over another that requires considerable manual effort. Once the data defining the model is available, variations from the original input should be possible to clean up noise introduced by automated methods, vary shape parameters obtaining similar but individual members of an object class, or simply correct designer input errors.

Computer graphics has frequently focused on the algorithmic side of the display task. There are costs associated with the display algorithms used to visualize each type of object model. Visualization cost can be as low as that of simple linear traversal and wireframe display of polyhedral objects, or as

costly and complex as ray-traced shaded renderings of algebraic surfaces. The fundamental observation is that display cost is dependent on the object model; computer graphics history is, in a large part, the study of procedures to display a particular class of object models. As in many other computer science algorithms, one pays in time for increasing generality of function. Thus the cost of ray tracing also entails the power to display a wide variety of object model types; at the other end of the spectrum, a simple depth buffer algorithm is highly effective for real-time display provided one is restricted to polyhedral models. Finally, the display cost is also dependent on image quality issues; for example, high resolution or animated images will require better models (and usually more detailed models), visualization studies might require particular object surface properties for advanced lighting models, etc.

Models are used for other purposes besides display, however, and the relative costs and complexity of these used must also be considered when selecting a model type. Two common uses are the transformation of an object in position, size, or shape, and the measurement of object geometric properties such as volume. These uses have costs associated with the geometric algorithms but the standard complexity measures must be weighed against the expected use. For example, the more efficient computational algorithm in the general case may have constants which make it less satisfactory in the expected majority of cases: a model consisting of arbitrary polygons will have less scan conversion efficiency than one composed entirely of triangles. Also, a frequently repeated algorithm should be made more efficient than one used once. Thus a simpler but theoretically costlier algorithm may be more efficient when multiplied by expected use. When programmer time is factored in, the costs of a graphical algorithm become a complex mix of special case considerations, overall efficiency, and programmer efficiency.

Certainly one important issue in object modeling is effectiveness: whether the model actually represents the desired phenomena and provides computational analogs of the appropriate visual or physical properties. Poor visual quality may be (though is not identical to) poor modeling. Thus the visual effectiveness of a stick figure of a human is compromised by the zero thickness model of body segments. On the other hand, a fractal model of a mountain may not tell us anything about the formation process of real mountains, but the visual effect and the statistical properties of the resulting surface may be quite convincing.

The complexity of a model may be judged better as a trade-off between the number of primitives and the inherent complexity (without being circular in reasoning) of the individual primitives. Thus at one end of the scale are objects which can be characterized as a multiplicity of simple forms; at the other end are objects which seem inherently complex. For example, machine parts, buildings, and other man-made objects seem to be mostly a multiplicity of simple forms, while biological and other natural structures (mountains, textures) are inherently complex.

Often object data will be available naturally in one form yet needed in another. This requires conversion from one form to another. The input may be formatted by an available input device (digitizer, laser scanner, tomographic images, manual drawing input, etc.), but the requirements of the application may dictate conversion to another form for more convenient display or computation. For example, surface point data may be approximated by polygons or curved surfaces to permit visual surface rendering in workstation hardware; constructive solid geometry models might be converted to voxels for volume computation.

7 Models and Rendering

It is important to understand the difference between models and rendering. Models describe the object and its attributes such as shape or geometry, color, reflectivity, transmittance, surface smoothness, and texture. Shade trees may be used to ascribe different visualization algorithms to different parts of a model. But it is the rendering algorithm itself which transforms the model to a screen-based view from a given camera position, projects the 3D data into display coordinates, determines the visible portions of the scene, and converts the object properties into pixel values in the context of light sources, atmospheric effects, anti-aliasing, image compositing, and color models.

8 Operations on models

We will ignore most of the rendering issues as they are discussed elsewhere in this Tutorial. We proceed to discuss the operations appropriate to object models. These include transformations, change of detail, measurement, combination, deformation, and display.

Transformations form the basis of most modeling systems. These include the well-known geometric transformations of translation, rotation, dilation (scaling), and reflection.

Change of detail is the process of adding or removing information to produce a finer grained representation (for better visual effect) or simplify other operations (for computational efficiency). Usually some interpolation method is used to augment detail: for example, fitting curved surface patches to point data to produce smooth continuous approximations to the expected original (partly-sampled) data. On the other hand, averaging or subsampling of the data may be used to reduce detail. Often models are grouped into hierarchic structures to simplify their organization: a model consists of sub-models, each of which consists of sub-models, etc., down to the level of one or more primitive types. An alternative hierarchic structure may be formed by defining the sub-models to be the same object as the parent object, only at a lesser degree of

detail. During model rendering, the appropriately detailed model is selected for display based on the projected size of the entire model on the display surface. During animation, as the size changes, the models from two detail levels may be visually blended with each other.

A number of measurements may be made on object models. The topology may be computed to determine whether the model is totally connected or consists of separate components. The presence of topological holes can be determined. The overall consistency of the object may be tested; for example, in surface-based models it is desirable to utilize topologically well-defined Euler operators to insure the validity of the object models being interactively designed.

Other useful or common operations include point-to-point distance measurement, computation of surface area or volume, and determination of surface tangents or normals for the rendering process.

Object models may often be combined to form new and more complex objects. The constructive solid geometry system is in fact based on Boolean combination as the fundamental operation. The combination operations are union (everything that is in either object), intersection (everything that is in both objects), difference (everything that is in one object but not the other), and symmetric difference (everything that is in either object but not in both). Other combination operations are the cut or slice which exposes the interior to view, and the cover operation which defines the relationship of one object to another in the finished image. The covering is an image property which does not necessarily stem from a real 3D model motivation.

There are a number of ways in which an object may be deformed. Deformation enriches the selection of object models without resorting to generation of totally new data sets. Deformation operations include skew, stretch, fold, and perturb (e.g. randomly, stochastically, or fractally).

The display operation is the most visible operation on an object model. There are many rendering algorithms, many of which are discussed elsewhere in this Tutorial: wire-frame, visible line, visible surface, ray casting, and ray tracing.

9 Representation structures

As we remarked earlier, the representation structures used for an object model may be either declarative or procedural. In a declarative representation, the model is explicitly embedded in a standard computational data structure. In a procedural scheme, the model is embedded into any convenient computational procedure, such as a formula, implicit equation, or arbitrary code. In the former, data is retrieved by search, indexing, or pointer chasing; in the latter, data is obtained by invoking a procedure with passed parameters or sending a message to an object which then executes a response.

10 Model classification

The major concern of the rest of this discussion is a taxonomy of models. We classify models into two broad categories: boundary schemes and volumetric schemes. In a boundary representation the surface of the object is approximated by or partitioned into (non-overlapping) 0-, 1-, or 2-dimensional primitives. We will examine in turn representations consisting of primitives formed by points, implicit surfaces or algebraic equations, polygons, fractals and graftals, and curved surface patches. In a volumetric representation the 3D volume of the object is decomposed into (possibly overlapping) primitive volumes. Under volumetric schemes we discuss voxels, octrees, constructive solid geometry, specialized (single primitive) systems, potential functions, and particle systems.

10.1 Surface and boundary models

The simplest surface model is just a collection of 3D points. They may be organized in a simple list, or may be more highly structured as contours, slices, or sections. Surfaces represented by points require a fairly dense distribution of points for accurate modeling.

10.2 Implicit surfaces / Algebraic equations

These surfaces are defined as the solutions to algebraic formulas. One familiar class of such surfaces are the quadrics. In general, the surface is the solution to an equation such as $F(x, y, z) = 0$ and numerical techniques or search are required if F is not analytic.

10.3 Polygons

Polygonal (polyhedral) models are one of the most commonly encountered representations in computer graphics. The models are defined as networks of polygons forming 3D polyhedra. Each polygon (primitive) consists of some connected vertex, edge, and face structure. There are a variety of data structures possible. Groups of polygons are stored in lists, tables, or linked structures to facilitate traversal of connected faces, the edge network, etc. The polygons are sized, shaped, and positioned so that they completely tile the required surface at some resolution. Polygon models are relatively simple to define, manipulate, and display. They are the commonest model processed by workstation hardware and commercial graphics software. In general polygons are best at modeling objects meant to have flat surfaces, though with a large enough number of polygons quite intricate and complex objects can be represented. "Large enough" may mean hundreds of thousands of polygons!

10.4 Fractals and Graftals

A limitation of the surface point and polyhedral models is the necessity (usually) of explicitly storing all the requisite data. If the objects to be modeled are highly irregular or complex, such as mountains or clouds, their statistical shape properties may be used to better advantage. Fractals and graftals create surfaces via an implicit model that produces data when requested. The surfaces are frequently based on point or polygonal models that are decomposed into finer and finer detail as the requirements of the display dictate. Fractals use special "random" processes to achieve a naturally motivated statistical irregularity such as demonstrated in rocks, mountains, clouds, coastlines, etc. Graftals use deterministic processes to model more repetitive patterns such as trees, leaves, snowflakes, etc. True fractals have a self-similarity property, where the frequency distribution of shape is supposed to be the same no matter what scale is chosen for examining the object. In practice, there are several variants on the true fractal definition.

10.5 Curved surfaces

Since polygons are good at representing flat surfaces, considerable effort has been expended determining mathematical formulations for true curved surfaces. Although implicit surfaces may exhibit true and continuous curvature, the need to solve equations is costly and usually undesirable. Most curved surface object models are formed by one or more parametric functions of two variables (bivariate functions). Each curved surface is called a patch; patches may be joined along their boundary edges into more complex surfaces. Usually patches are defined by low order polynomials (typically cubics) giving the patch easily computed mathematical properties such as well-defined surface normals and tangents, and computable continuity conditions between edge-adjacent patches. The shape of a patch is derived from control points or tangent vectors; there are both approximating and interpolating types. The former take the approximate shape of the control vertices; the latter must pass through them. There are numerous formulations of curved surfaces, including: Bezier, Hermite, bicubic, B-spline, Beta-spline, polynomial, rational polynomial, cardinal splines, composite splines, etc.

10.6 Volume and CSG models

The first volumetric model we examine is the voxel model. Here space is completely filled by a tesselation of cubes or parallelopipeds called voxels (volume elements). Usually there is a density or other numerical value associated with each voxel. Storing a high resolution tesselation is expensive in space but simple in data structure (just a large 3D array of values). Usually some storage optimization schemes are required for detailed work (1K x 1K x 1K spaces). Special

techniques are needed to compute surface normals and shading to suppress the boxiness of the raw voxel primitive. Voxel data is commonly obtained in the medical domain; it is highly regarded for diagnostic purposes as the 3D model does not speculate on additional data (say by surface fitting) nor suppress any of the original data however convoluted.

10.7 Octrees

Octrees are one of the data structures used for volumetric models that tesselate a given 3D space. The original volume, say a cube, is partitioned into 8 cubes if it is non-empty. Recursively, each sub-cube is partitioned whenever non-empty, until some minimum size element is reached. Since empty cubes are not sub-divided, the storage space efficiency is increased. The major use of octrees appears to be an indexing scheme for access efficiency in a large 3D array.

10.8 Constructive solid geometry

One of the most efficient and powerful modeling techniques is constructive solid geometry. Unlike the voxel and octree schemes, there is no requirement to regularly tesselate the entire space. Moreover, the primitive objects are not limited to (uniform) cubes; rather there are any number of simple primitives such as cube, sphere, cylinder, cone, half-space, etc. Each primitive is transformed or deformed and positioned in space. Combinations of primitives or of previously combined objects are created by the Boolean operations. An object therefore exists as a tree structure which is "evaluated" during rendering or measurement.

10.9 Specialized (single primitive) systems

The generality of the constructive solid geometry method – with its multiplicity of primitive objects and expensive and slow ray-tracing display method – is frequently reduced to gain efficiency in model construction, avoid Boolean combinations other than union, and increase display speed. The idea is to restrict primitives to one type then design manipulation and display algorithms to take advantage of the uniformity of the representation. Voxels might be considered such a special case, where the primitives are all coordinate axis aligned and integrally positioned cubes. Other schemes are possible, for example, using ellipsoids, cylinders, superquadrics, or spheres.

Ellipsoids have been used to model cartoon-like figures. They are good for elongated, symmetric, rounded objects. Unfortunately, the display algorithm is nearly the same as the general ray-tracing process. One unusual application of (overlapping) ellipsoids is to model terrain, clouds, shrubbery, and trees. In these cases, the convincing visual effect of semi-solidity is achieved by statistical transparency which has a low rendering cost.

Cylinders have also been used to model elongated, symmetric objects. An application that uses cylinders plus spheres with a special-purpose display algorithm is ball and stick molecular modeling.

Superquadrics are a mathematical generalization of spheres which include an interesting class of shapes within a single framework: spheres, ellipsoids, and objects which arbitrarily closely look like prisms, cylinders, and stars. Simple parameters control the shape so that deformations through members of the class are simple and natural. Superquadrics are used to model man-made objects, but when overlapped can give the appearance of faces and figures.

Spheres as a single primitive form an intriguing class. They are the only modeling primitive that is isotropic: that is, identical in appearance from any view. Moreover, spheres have a simplicity of geometry that rivals that of simple points: just add a radius. There are two methods of rendering spheres. They can be drawn as fully 3D objects in which case some scan conversion tricks can be used to simplify the generation of successive object points (basically a generalization of incremental circle drawing algorithms). An alternative display method treats the spheres as if they were "scales" on the modeled object; in this case a sphere is rendered as a flat shaded disk. With sufficient density of overlapping spheres, the result is a smoothly shaded solid which models curved volumes rather well. A naturalistic human figure may be done this way.

10.10 Potential functions

An interesting generalization of the sphere model is to consider the volume as a potential function with a center and a field function that decreases monotonically (by an exponential or polynomial function) from the center outward. There is no "radius" or size of the potential function; rather, the size or surface is determined by setting a threshold value for the field. What makes this more interesting is that potential functions act like energy sources: adjacent potential functions have overlapping fields and the resultant value at a point in space is in fact the sum of the fields active at that point. Thus adjacent fields blend smoothly, unlike the "creases" that are obtained with fixed radius spheres. Recently, directional dependence and selective field summation across models have been added to create "soft" models that blend with themselves but not with other modeled objects in the environment. Potential functions were originally used to model molecules, since atoms exhibit exactly this form of field behavior. Other uses for potential functions are real and imaginary organic forms including human and animal figures.

10.11 Particle systems

Generalizing spheres in a different direction, particle systems reduce the sphere to a zero radius. A volume is therefore characterized by a set of (usually moving) particles which indirectly define the space. Each particle has its own color,

path, history, and lifetime. Their motion is typically controlled by probabilistic algorithms. Particle systems have been used to model fire, gases, explosions, fireworks, and grassy fields.

11 Where do models come from?

There are numerous methods for generating 3D surface or volume models. We briefly examine manual digitization, semi-direct data acquisition, and algorithmic methods.

Manual digitization is the construction of a model from 3D coordinate data measured and typed in or directly traced from plans, models, or the real thing. Several forms of digitizers exist to aid this task. Often data from two or more views is used so that 2D data may be extrapolated into 3D. If two views are taken with properly positioned and calibrated cameras, manual selection of corresponding points on the two views can be input to a 3D reconstruction formula. This is the basic operation in photogrammetry.

If manual input is too tedius, or the model too complex, semi-automated data acquisition methods may be used. Principals among such methods are direct 3D coordinate input via 3D digitizers, laser scanning to obtain gridded range (hence depth) information, voxel reconstruction from multiple planar (tomographic) images, and direct (video) image analysis. The last method is still in relative infancy, but computer vision techniques can be used to intelligently reconstruct various simple geometric models based on polyhedra, curved surfaces, spheres, cylinders, and superquadrics.

Algorithmic and interactive design methods form a large class of model building techniques. Typically an interactive graphical editor is used to build a model within the particular modeling system representation. For example, polyhedral models are pre-eminent in computer-aided design applications, with curved surfaces and constructive solid geometry models becoming widely available. Other interactive model editors have been constructed for the other model types. Often, an algorithmic method can assist the design process by enforcing constraints on the model primitives. For example, using Euler operators will insure that a polyhedral model is topologically well-defined (though it may still be odd in a geometric sense). Actual constraint operators may insure that features within a model are parallel, tangent, horizontal, vertical, non-intersecting, etc. Finally, some algorithmic methods can be used to interpolate or generate data from forms more easily input; for example, by creating reconstructed surfaces between parallel contour slices of surface points, by fitting curved surfaces to control vertices or tangency constraints, or by using force or energy methods to cause model pieces to self-assemble.

A Glossary for Modeling and Animation

Norman I. Badler [*] Andrew S. Glassner [†]

Adaptive sampling Evaluating an image coarsely at first, and then more finely in regions with detail. Causes for taking more samples include complex shapes, textures, and shadows. See super-sampling.

Aliasing Artifacts in an image generated when we only use a finite number of equally-spaced samples, such as at pixel centers. The artifacts include banding, Moirè effects, jaggies on polygon edges, and regular patterns that don't belong.

Algebraic Surface A surface defined by an equation. The surface contains all points which evaluate to a particular value of the equation, usually zero. Algebraic surfaces are a class of implicit surfaces.

Ambient The amount of illumination in a scene which is assumed to come from any direction and is thus independent of the presence of objects, the viewer position, or actual light sources in the scene.

Animation The process of creating and recording images which change over time. Though often interpreted as implying only two-dimensional image changes, it may be applied in general to any model or scene changes in three dimensions as well.

Anti-aliasing The process of reducing, removing, or avoiding aliasing artifacts. See aliasing.

Articulation The ability of one part of an object to move with respect to another, such as a hinge or sliding joint.

Background A fixed image which is interpreted as existing infinitely far from the observer. It is displayed when no object surface falls into an image sample.

Bend Deforming a shape along a line.

[*] Computer and Information Science Department, University of Pennsylvania, Philadelphia, PA 19104 USA

[†] Microsoft Research, One Microsoft Way, Redmond, WA 98052

Binary Space Partitioning (BSP) A method for dividing up a 3D space with a set of planes. For any observer and plane, each object in the environment either straddles the plane, is on the same side as the observer, or on the other side. Objects on the same side of (and in front of) the observer will block visibility of objects on the other side.

Bivariate patch A vector-valued function of two variables, typically called u and v. These parameters generate some locally two-dimensional surface in space. Often the parameters are assumed to lie in the range [0,1]. See patch.

Blob An implicit surface. Blobs are usually radially symmetric Gaussian functions.

Boolean operation An algebraic combination (union, intersection, or difference) of two geometric solids.

B-Rep See Boundary representation.

Boundary representation The geometry of an object as described by its surface. The surface of the object is approximated by or partitioned into (non-overlapping) 0-, 1-, or 2-dimensional primitives (points, curves, and surfaces).

BSP See Binary Space Partitioning.

B-Spline A type of spline, where the curve is influenced by the control points, but generally does not pass through them. See Splines.

Bump mapping A technique for faking fine-level geometry on a surface, such as the bumps on an orange rind. Rather than actually create many little pieces of surface to model the bumps, a texture map is placed on the surface which describes how the surface normal would vary if the shape really had bumps. By shading with these imaginary normals, the surface appears to be bumpy. Drawbacks to the technique are related to the fact that only small bumps can be faked this way: the silhouette is not changed, bumps don't occlude other bumps, and usually the bumps don't cast shadows.

CAD Computer-Aided Design. The process of manually creating geometric object models which meet desired design criteria.

Catmull-Rom Spline A type of spline, where the curve is influenced by the control points, but generally does not pass through them. See Splines.

Cel animation Animation produced by stacking multiple 2D drawings (called *cels*), each containing a fragment of a scene or character on a transparent background. In a physical environment the cels are lit and photographed

to create one frame of animation. Cels are basically a labor-saving device. If a character is standing still and talking, then a cel of the character without a mouth might be placed over a background. Then for each frame a cel containing the appropriate mouth image is placed on top of the stack, avoiding the necessity to re-draw the entire character for every frame. Often there are several independent planes of images which may be superimposed in a so-called "two and a half-dimensional" animation.

Clipping planes Boundary planes in the world coordinate space which delimit the portion of the geometric model which will be rendered. Typically top, bottom, and side clipping planes are based on the window and camera position. The front and back (or *hither* and *yon*) clipping planes are used to select the object depth limits to remove undesired foreground (closer) or background (further) objects or slice the object perpendicular to the view to expose inner detail.

Compositing The process of creating an image by combining multiple independent images. A *matte* describes how much each point of each image contributes to the final picture. To animate a spaceship, for example, one might render the spaceship, and then composite it over a background of stars. The ship would completely replace the starfield, but some of the stars might be visible through the exhaust. In animation, the background and stationary objects are rendered once, while changing or moving objects are rendered on a per frame basis and combined with the static part of the image.

Constraints Values, relationships or conditions which are to maintained while other changes are being made. For example, a constraint can hold a line horizontal, keep two objects a constant distance from one another as they moving, or can cause one object to be attracted to another.

Constructive Solid Geometry (CSG) A model representation based on a set of primitive geometric shapes which may be transformed spatially and combined in a tree-structured fashion by Boolean operations. The resulting model is a tree with the primitives as the leaves, Boolean operations as the non-leaf nodes, and edges as transformations.

Contour Typically a curve in space. Common contours are those used as the cross-section for swept surfaces, and the silhouette of some object from a particular point of view.

Control Point One of the points that influences the shape of a spline or patch. See splines, patches.

Concave See Convex.

Continuity When two patches share an edge, that edge may be more or less noticeable. The *continuity* of the edge across the patches is a mathematical measure of how smooth the edge is. If the edge has *0-degree* continuity, then there are simply no holes, but there may be a crease. If the edge has *first-degree* continuity, then the first derivative of the surface is continuous across the edge. This derivative may be measured either *parametrically* or *geometrically*; the former is derived from the natural mathematical description of the patch, and the latter comes from the apparent (or visible) nature of the patches. Higher degrees of continuity refer to ever-smoother blends of the surface across the edge.

Convex A property of a shape in any dimension. A shape is *convex* if you can pick any two points in the shape, and connect them with a straight line that never goes out of the shape. In 3D, a sphere is convex, but a wineglass is not (pick a point in the bowl and a point on the base; the line between them is not entirely within the wineglass). A shape that is not convex is concave.

Convex Hull The smallest polyhedron that encloses a given object. The convex hull for a polygon can be created in 2D by drawing the polygon on a wooden board, and then driving in a nail at each vertex. Stretch a rubber band so it encloses all the nails, and then release it; the shape formed by the rubber band is the convex hull.

CSG See Constructive Solid Geometry.

Database amplification The notion that simple sets of parameters stored in a data structure or database may expand into quite complex objects by the application of suitable procedural processes. Examples are fractal and graftal shapes which require only a few numbers or syntactic rules to enable the generation of intricate and detailed models.

Data structure A computerized method of storing information such as a list, array, graph, network, etc.

Decomposition Breaking an object into smaller or simpler components, typically to facilitate the algorithmic processing of the original object by more general procedures.

Depth cueing The process of reducing the apparent brightness of an object the further away from the viewer it is positioned. This often enhances the perception of depth. The idea is that one is viewing the scene under foggy conditions, where the contrast diminishes with distance. If the background is black, this diminished contrast is equivalent to reduced brightness; if the background is gray, it is reduced saturation of the color.

Difference A Boolean operation over two objects where the resulting object is the set of points in the first object that are not contained in the second object. Difference is also called *subtraction*. This is commonly used to drill holes in a part by subtracting a cylinder from the part. See symmetric difference.

Diffuse Light which is reflected equally in all directions from a point on the surface of an object. The brightness of the surface does not, therefore, depend on the observer's position relative to the surface point.

Digitization The process of building a geometric model from existing drawings, physical objects, or models of objects. The three-dimensional coordinates are sensed, located, typed, or otherwise obtained from the source.

Dilation A scaling geometric transformation. Scaling of a point coordinate (multiplication by a constant) can occur independently in each dimension.

Dithering The process of trading spatial resolution for enhanced intensity or color resolution by distributing fewer shades over a region larger than a single pixel. The eye performs the spatial integration to produce the additional shades or intensities. Care must be taken to avoid producing distracting spatial patterns.

Dynamics Animation by the specification of forces and torques applied to masses, rather than their positions, velocities, and accelerations.

Easing A reduction in the acceleration or deceleration of a motion to present a smoother, more continuous movement. Also used to blend two motions together.

Environment Map A picture of the environment surrounding a model. It is usually used to approximate reflections off the model.

Field of view The solid angle across the viewing pyramid at the apex (center of projection) where the camera is located.

Fractal A geometric structure having the property that its frequency distribution is the same no matter what magnification (scaling) factor is applied to it. Examples typically cited are coastlines and clouds. In practice the term is also applied to any curve or surface obtained by the stochastic (random) perturbation of points on or in the surface as the shape is subdivided to (sub-)pixel size for rendering. Technically, a fractal is a mathematical construction that is infinite; any computer models based on these ideas are just approximations of true fractals.

Geometric editor An interactive program which permits the creation and modification of shapes by distinct operations on the geometric data structure representing an object, for example, instancing a primitive shape, transforming it, combining it with other shapes, etc.

Goniometric diagram This is a diagram that gives the intensity and color of the energy leaving a point light source for every direction in which light can leave the source. The diagram may be used with approximations for light sources with finite size.

Graftal Any curve, shape, or surface obtained by the deterministic or syntactic perturbation of points on or in some starting primitive as the shape is subdivided to (sub-)pixel size for rendering. Often graftals are generated by a *shape grammar*.

Hierarchy A tree in which nodes typically represent objects and edges represent relationships. For example, a human body could be represented by a hierarchy with the torso at the top (or *root*), and five children representing the arms, legs, and head. Each arm could be a smaller hierarchy of upper arm, lower arm, and hand; the hand is a hierarchy of five fingers, and so on.

Hither Clipping Plane See clipping planes.

Implicit surface A surface described by an equation. The surface contains all points which evaluate to a given number when plugged into the equation; usually this value is zero. Often the solutions must be found by numerical search techniques.

In-betweening See interpolation.

Interpolation The process of filling in information from a set of values. Interpolation can be used to generate plausible surface points for sparse data or for curves or surfaces defined with a small set of control points. It is also a way to generate positions for objects between key frames in an animation.

Intersection A Boolean operation over two objects where the resulting object is the set of points which lie in both objects simultaneously.

Key frames Cel animation images drawn by the animator which are interpolated algorithmically by the computer to generate inbetween images at the desired frame rate.

Key parameters Values of motion or other attribute parameters which are specified by the animator. Inbetween parameter values are typically generated by spline curves.

Kinematics Motion specification by position, velocity, and acceleration, rather than by forces. Usually applied to the manipulation of mechanisms involving articulated parts and various joints and constraints between the parts.

Level surface All solutions to the implicit equation $F(x) = k$ for some constant k and multidimensional vector x.

Light source The colloquial name for a luminaire.

Luminaire An object which emits light on its own, even if no other light is striking it from elsewhere.

Local coordinate system A reference frame positioned within or on an object to facilitate its definition, manipulation, and placement in a scene. The local coordinate system moves along with the object.

Matte A two-dimensional region (not necessarily connected) of an image which indicates where one image is to be composited with another.

Mip-Map A pre-processed form of a texture which can increase rendering speed. "Mip" stands for *multum in parvo*, Latin for "many things in a small place". Similar to the sum table.

Octree A data structure for a spatial partition which divides space into eight equal cubes (*octants*). Each cube is recursively subdivided as needed to the level of detail of the model or the pixel size.

Orthogonal A synonym for *perpendicular*.

Orthogonal projection The view created by placing the camera at an infinite distance from the scene. The view direction creates a viewing parallelopiped whose cross-section is determined by the window. Objects of the same size at different depths from the camera appear the same size in the image.

Particle system A representation which consists of (large) sets of individual points under algorithmic or stochastic control. The points have color, a trajectory, and a history. They may be created, destroyed, or split. The history may be used in a single image to create a curvilinear shape, or in an animation to create a moving point.

Patch Almost any 3D curved surface may be thought of as a patch. Typically they are small bits of surface that are created by a mesh of *control points*, which influence the patch in the same way that they influence a spline. Patches are usually connected to preserve different types of continuity. See control points, continuity.

Paths The value of a parametric function over time. Typically used to describe the spatial position of an object, but also applicable to its orientation, or even attribute values.

Perspective projection The view created by placing the camera at a finite position (the center of projection) within the scene. The camera forms the apex of a viewing pyramid whose cross-section is determined by the view direction and the window or field of view. Objects of the same size

at different depths from the camera appear in different sizes in the image: the further objects being foreshortened and hence appearing smaller.

Polygon A closed and connected sequence of vertices and edges. Usually considered planar and simple (non-intersecting). Polygons may also have internal edge loops (creating holes) or multiple components.

Polyhedron A 3D solid whose boundary is made of polygons.

Potential function Typically an exponential function with a center point and a value which decreases as a function of radius from the center. A level surface of a potential function is a sphere of some radius. By analogy to electric fields, adjacent potential functions may be summed so that the level surface of a set of potential functions is a surface of constant "charge" and therefore represents a smoothed combination rather than hard-edge sphere intersections. See algebraic surface and blob.

Primitives The simplest unit of geometric object represented in a data structure. The geometric, measurement, and rendering algorithms operate over primitives such that the results can be combined to apply to more complex objects consisting of admissible combinations of primitives.

Procedural representation The geometric model is embedded into any convenient computational procedure, such as a formula, implicit equation, or arbitrary code. The procedure may generate data structures (e.g. a procedure to create a cylinder of given length and number of sides as polygons), provide data values (e.g. from the evaluation of a function describing a surface), or return parameter values (e.g. a wave model where the outputs are control points for curved surfaces representing portions of the wave). Very complex models such as mountains, and very regular models such as a bookshelf of books, are usually good candidates for a procedural model.

Quadratic surface A surface defined by algebraic equations with variables raised at most to the power 2.

Quadric surface See Quadratic surface.

Quadtree A partition of the plane into four quadrants, each of which may be recursively subdivided into quadrants to the desired level of object detail or pixel size.

Radiosity A shading process in which the visibility of any portion of a surface of an object is assessed relative to every other surface. Because all the illumination is pre-computed once, the database can be stored as colored primitives that represent their illumination, and real-time hardware may be used to simply display the primitives.

Ray casting The process of determining which object in a scene is first encountered by a ray emanating from the camera view location and passing through a pixel on the display screen. From the list of ray-object intersections the first visible intersection is chosen. The surface normal at this point is used to determine the shading value.

Ray tracing The process of determining the shade of a pixel in a scene consisting of arbitrary objects, various surface attributes, and complex lighting models. The process starts like ray-casting, but each ray is followed as it passes through translucent objects, is bounced by reflective objects, intersects objects on its way to each light source to create shadows, etc.

Reflection A scaling geometric transformation where one or more of the coordinate multipliers are negative. Also the effect of light bouncing off an object. See reflectivity.

Reflectivity A surface attribute of an object which causes incoming light to be reflected only at the angle opposite to the incidence angle about the surface normal.

Rendering The process of taking a geometric model, a lighting model, a camera view, and other image generation parameters and computing an image. The choice of rendering algorithm is dependent on the model representation and the degree of realism (interpretation of object and lighting attributes) desired.

Rotation A geometric transformation which causes points to be rotated about some axis through the origin by a given angle.

Sampling The process of selecting a finite number of places to probe or sample a continuous function or scene model in order to produce a dense enough set of values to produce a reasonable discrete approximation.

Scan-line algorithm A rendering technique which computes the image one scan line at a time taking advantage of the minimal changes from one scan-line to the next in an image. The algorithm reduces a three-dimensional visibility problem to a two-dimensional problem per scan-line.

Scripts Time dependent statements of events or action. The actions are motion parameters applied to objects. Often a script resembles a programming language but the timing of events may also be visualized in a graphical form.

Shade tree A custom shading procedure for a particular object. This procedure may use all sorts of information from the object and the environment to create complex surface descriptions, including simulated reflections and shadows. A shape tree may also be used to describe the output of a luminaire. See luminaire and goniometric diagram.

Shading model The algorithm used to determine the color of light leaving a surface given a description of the light incident upon it. The shading model usually incorporates the surface normal information, the surface reflectance attributes, any texture or bump mapping, the lighting model, and even some compositing information.

Skew A geometric transformation which causes points to be translated in one or more dimensions while remaining fixed in another. The amount of translation is a multiple of the value of the fixed coordinate.

Slice The process of dividing a model by a plane so that one portion may be removed to view the interior structure. It is also sometimes used to describe the contour of the object obtained in the cutting plane itself.

Solid Texture A texture which is defined at every point in space. An object textured this way appears to be "carved out" of a volume of material with this texture. Common examples include wood and marble.

Spline A curve, usually defined by a series of points called control points. In shipbuilding a spline is a long thin piece of wood suspended on two supports. At different places along the board weights are placed to create gentle curves; heavier weights exert more of an influence. Mathematical splines were originally developed to simulate this physical construction. Thus one speaks of the *weight* associated with a control point.

Specular The component of illumination seen at a surface point of an object which is produced by reflection about the surface normal. It depends on the observer position, whereas the diffuse component does not. Often this appears as a "highlight."

Stochastic sampling A technique for anti-aliasing. The basic idea is to replace the regular sampling grid (such as one sample in the center of each pixel) with a more random, or stochastic, set of sample locations. The result is that regular aliasing artifacts such as jaggies are replaced by fuzzy, noisy edges. When enough samples are used the noise smooths out and none of the regular aliasing artifacts appear. The advantage of this technique is that it can eliminate the regular aliasing artifacts created by every regular grid.

Stretch A geometric transformation which deforms a object locally. It is usually applied to the control parameters of a curved surface.

Sum Table A pre-processed form of a texture which can increase rendering speed. Similar to the Mip-map.

Super-sampling The process of sampling a scene at a higher frequency than the pixel spacing. The resulting excess samples are averaged or filtered to

determine the actual pixel value. In contrast to adaptive sampling, supersampling refers to taking many samples everywhere in the image, rather than just where they are needed.

Surface normal The vector which is perpendicular to the object surface and outward facing. Surface normals are one of the principal determiners of surface shading.

Surface tangent The plane which is tangent to the object surface. The tangent is primarily used when discussing the continuity of curved surface patches across patch boundaries.

Surface of Revolution A 3D shape created by a 2D curve that is rotated around an axis. Such shapes may be created on a lathe. Examples include candlesticks and wine glasses.

Swept Surface A 3D shape created by a 2D curve (called the *contour*) that is swept along some 3D curve (called the *path*). The contour may be scaled or otherwise transformed as it moves along the path. For example, a circular contour may be swept along an S-shaped path to make a snake.

Symmetric difference A Boolean operation over two objects where the resulting object is the set of points in the first object that are not contained in the second object and the set of points in the second object that are not contained in the first. See difference.

Texture A two- or three-dimensional pattern which is applied to the object surface during the shading computation. Textures may determine color, reflectivity, transparency, material, or any other factor that controls how a shape looks. A *displacement* texture can even mildly distort a surface. The advantage to texture is that the complexity of appearance does not have to be explicitly modeled in the object geometry.

Tesselation A partition of the plane or space into one or more non-overlapping primitives. The primitives are usually regular in size and shape. Common examples in 2D are squares and hexagons. In 3D, cubes are convenient.

Tile To cover a (possibly non-planar) surface with planar polygons. Also, a tile sometimes refers to a texture pattern over a square polygon.

Topology The mathematical notions that describe the underlying structure of an object without reference to its particular shape. Two meshes of rectangles of equal size have the same topology, regardless of the locations of their vertices. Topologists are fond of noting that there is no difference between a donut and a coffee cup; a flexible model of either one can be deformed into the other by stretching and pulling, without any ripping or cutting.

Transmittance A property of a surface describing the fraction of light that will pass completely through it. Thus a transmittance of 0 is opaque, 1 is invisible, and values inbetween offer various degrees of translucency.

Translation A geometric transformation which causes points to be moved by constant displacements in each coordinate.

Union A Boolean operation over two objects where the resulting object is the set of points which lie in either object.

View direction The direction in which the camera is pointing. It is the perpendicular to the viewing plane on which the image is formed.

View up direction The world direction which when viewed by the camera will appear to be vertical (up) in the image.

Viewport The sub-area of the screen or image formation device into which a graphical rendering or other drawing is placed.

Visible surface That part of an object which is visible from a given point of view.

Visible line The process of rendering or drawing only linear features (lines) to show at least only those edges or contour lines of surfaces which are in fact visible to the camera.

Voxel A three-dimensional volume element, usually a cube or box. It has a value (or density) and six sides and represents a finite sized point in a regular decomposition of space.

Window A rectangular subregion of the viewing plane which defines the top, bottom, and side clipping planes relative to the camera position.

Wire-frame The drawing of a model by tracing linear features such as edges or contour lines without attempting to remove invisible or hidden parts.

World coordinates The arbitrary coordinate reference frame in which object models are positioned to create a scene.

World projection A special background image built from the objects in the scene itself which is used to create convincing reflections on objects.

Yon Clipping Plane See clipping planes.

Z-buffer A rendering technique in which objects are scan-converted to pixel data and depth values and then inserted into a pixel array and a depth array. Where the object's pixel depth is less than that of the currently stored pixel/depth pair, the new data is written in, replacing the old. Advantages of the Z-buffer are that the objects need not be processed in

any special order, the image can be incrementally updated with additional objects at any time, and the scene can contain an unlimited number of objects.