PART STRUCTURE FOR 3-D SKETCHING

Alex P. Pentland
Artificial Intelligence Center, SRI International
333 Ravenswood Ave, Menlo Park, California 94025
and
Center for the Study of Language and Information
Stanford University, Stanford California 94038

ABSTRACT

Natural, efficient communication depends upon shared representations. Current 3-D graphics systems, however, use representations that are quite distant from that which people use. The result is that construction of 3-D models is much like programming: meticulous translation from the persons' internal representation to the machines' representation. We argue that a constructive solid geometry representation that allows stereotyped deformations and statistical specification closely parallels peoples' internal representation. Such correspondence allows fast, "natural" 3-D modeling; this is especially important in the initial stages the design process where a "sketching" capability is more important than the ability for precise control of details. We describe and evaluate an interactive system that uses such a representation. The system demands real-time interaction; to support this on 68020-class machines we develop a linear-time hidden line algorithm, so that the hidden-line calculation requires only slightly more time than is needed to draw the lines.

1 Sketching versus Detailing

The distinction between *sketching* and *detailing* is important in understanding how people create a 3-D model. For instance, engineers typically sketch a new part using paper and pencil, and then give the sketch to a draftsman who uses a CAD system to complete the detailed specification of the model. Similarly, animators sketch out scenes and actions before drawing careful renditions of the sequence. The reason that people standardly divide the design process into two stages — each employing its' own media — is that there are two conflicting sets of requirements: the initial design of a 3-D model (i.e., 3-D sketching) demands the ability for quick, general-purpose, and natural interaction, while the final drafting or rendering stage demands the ability for detailed, precise control.

Most current 3-D graphics systems have the wrong "control knobs" for the initial, sketching phase of the design process; that is, the things you would like to do when "roughing in" a 3-D model aren't usually easy to do. This makes things difficult; you have to approach the task of modelling a shape in a planned, methodical manner, much as a programmer approaches the problem of constructing a program[1] Because you have to carefully plan your interaction with the machine, both engineers and graphic artists still sketch shapes on paper before attempting to use a 3-D modeling system.

The use of paper for sketches and computers for final models is bad for exactly the same reasons that the use of paper for final models is bad: lack of flexibility, unneeded duplication of effort, no library of previous drawings, and so forth. In an attempt to address these problems we set out to develop a 3-D modeling language, user interface, and rendering system that is sufficiently "natural" and interactive that people would choose to sketch shapes on the *computer* rather than sketching them out on *paper*.

The idea, then, was to develop a tool that allows the user to *very* quickly build or modify a 3-D model; to replace the pencil and paper. A user would directly *sketch* 3-D form on the computer, playing with the shape until it looks right, rather than approaching the modeling task as one of entering a carefully predefined model into the computer. An engineer would quickly "sketch" a new part directly on the computer, playing with it until it satisfied him. An animator would "sketch" a scene and, Claymation-like, interactively modify the scene so as to step through key points in an action sequence. In both cases, once we are satisfied with this "sketch model," we can then invest the time to carefully fill out the models' details using a system that is specialized for that particular task.

We want, therefore, a tool that is not specialized to any one application domain but, like pencil and paper, is equally applicable to any 3-D modeling task. And further, like pencil and paper, we want this modeling tool to be generally *available*: i.e., cheap enough to sit one on everyones' desk, so that they will actually use it.

1.1 The Design of a Graphics System

We have implemented our solution to these problems in a system called SuperSketch (named for "sketching" and "superquadrics"), which provides an environment for interactively sketching and rendering 3-D models. The specific major design criteria for SuperSketch were:

(1) Representation: The system must have a communication metaphor (language) that closely matches the way people naively think about and discuss shape, to promote easy, natural communication between the user and the machine.

(2) Interaction: The system must have an interaction interface that allows users to attain a level of "effortless" interactive control similar to that of an engineer or artist sketching in pencil.

(3) Efficiency and Accessability: If it is to be truely useful, the system must be efficient enough to allow "real-time" line drawings and rapid full color renderings on a computer inexpensive enough to sit on everyones' desk; e.g., a Motorola 68020-class machine without additional hardware.

In the following sections of this paper we will discuss how we have sought to meet each of these design criteria.

2 Representation

The process of constructing and animating a 3-D model is a process of communication between the machine and the human operator. Because communication depends upon having a shared representation of the situation, the development of natural, "effortless" methods for constructing and animating 3-D shapes depends upon having a representation that is isomorphic to that which people use. When the representation used by the machine doesn't match the way the human operator thinks of