

Visit to Prof Norio Okino, Kyoto University, July 26, 1991

In the 1960's and 70's Prof Okino developed one of the first solid modelers based on constructive solid geometry, called TIPS. He still works on CAD/CAM and solid modeling but his main interest recently has been what he calls "bionic manufacturing."

Bionic manufacturing involves two elements: self-governing behavior and object-oriented hierarchical structures. Prof Okino has constructed a conceptual model of Computer-Integrated Manufacturing (CIM) that begins with all of society at the top and extends downward by subdividing object classes until the lowest levels are reached somewhere in a factory.

Each level is composed of elements that have the same structure in principle. These are called "modelons." Each modelon contains a common memory and a number of processes or methods that operate using that memory. In addition, a number of lower-level modelons are attached to this memory. Modelons are independent actors, like daemons or, in UNIX, processes. Each modelon looks out for the conditions under which it could run; if they are satisfied, the modelon runs and deposits its results in the common memory it is attached to. Modelons are also like methods in object-oriented programming in the sense that they can send messages to each other, activate each other, etc.

A major feature of this structure is, as said above, that each modelon acts autonomously. What the convergence problems of such a structure might be Prof Okino does not say.

Several topics are being studied under this structure. I was shown software demonstrations of two: a robot modelon that interacts with prismatic peg and hole modelons to determine where to grasp the peg for the purposes of putting it in the hole; and a hidden line removal algorithm in which modelons representing each of three prismatic parts inform each other of the locations of their vertices. We discussed a third for which there was no demonstration: a shaft modelon made of three shape features, each of which is represented by generic feature modelons. All the work is being done on Apollo Domain 10000 workstations.

1. Robot grasp planning: This is the recently completed work of Dr Watabe. In this system, there are several software modelons representing the parts, the robot, and the environment. The user requests a task, such as that the peg be put in the hole. The top level modelon broadcasts this request to the lower ones, all of which attempt to respond with a solution. Finding that they cannot, they broadcast in turn to any of their subelements whose responses they need or could use, in a divide and conquer approach. This is repeated at lower levels recursively. For example, the robot needs to know several sets of faces on the peg, such as parallel faces, parallel faces free-to-grasp before inserting, the same after inserting, and so on. The peg and hole modelons respond. I could not tell how the assembled state was established for

the purpose of identifying the free faces. Perhaps the user constructed it as a way of posing the problem.

There is no agenda structure in this search. Dr Watabe had not heard the terms "forward chaining" and "backward chaining" before. (These are common techniques in expert systems for solving the kind of problem he is working on.) Such searches commonly are not very efficient since there is no gradient to follow and no metric to score how close one is to finding a solution.

2. Hidden line removal: In this demo, three prismatic blocks intersect each other. Before the hidden lines are removed, one cannot see what is what, even with three views, because several of the edges appear parallel and the front - back optical illusion interferes. These facts make the demo more interesting but of course have nothing to do with the intrinsic difficulty of the problem. Each block has its own hidden line removal method and seeks information from the other blocks concerning where its lines enter or leave their boundaries. Information is passed around this way for several minutes before a solution appears on the screen.

Prof Okino points out that speeding up the algorithm or competing with existing algorithms is not the objective, but rather it is to understand modelons.

In this regard, it is interesting to review the discussion we had about the shaft made of three shapes: a plain cylinder, a conical cylinder, and a threaded cylinder, all coaxial. Each shape is supported by a generic feature that contains methods for drawing the shape, calculating its mass, and a process plan for how to make it. Presumably the process plan for the shaft is made by combining the process plans for the three supporting features.

I asked a basic question that underlies the problems in all process planning of this kind: what to do in regions where the plans touch or intersect and presumably interact? That is, how does one compose process plans from subplans? He agreed that this was a challenging question and replied that perhaps one must declare the shaft itself to be the primitive element. A student is beginning to work on the composition problem.

Prof Okino's reply indicates that there are many problems yet to be solved by this approach. The value of having generic elements at the leaves of the structures is clearly large and would give the approach considerable power. Requiring each specific shaft to have a representation of its own is not efficient. However, he is a software person rather than an engineer and is pursuing the structural issues first.

Prof Kimura notes that object-oriented structures are good for some kinds of data but not others, especially those that have strong interactions as well as, or instead of, hierarchical, decomposable structures. Mechanical design and manufacturing may not be separable enough to permit object-oriented approaches to cover them completely.