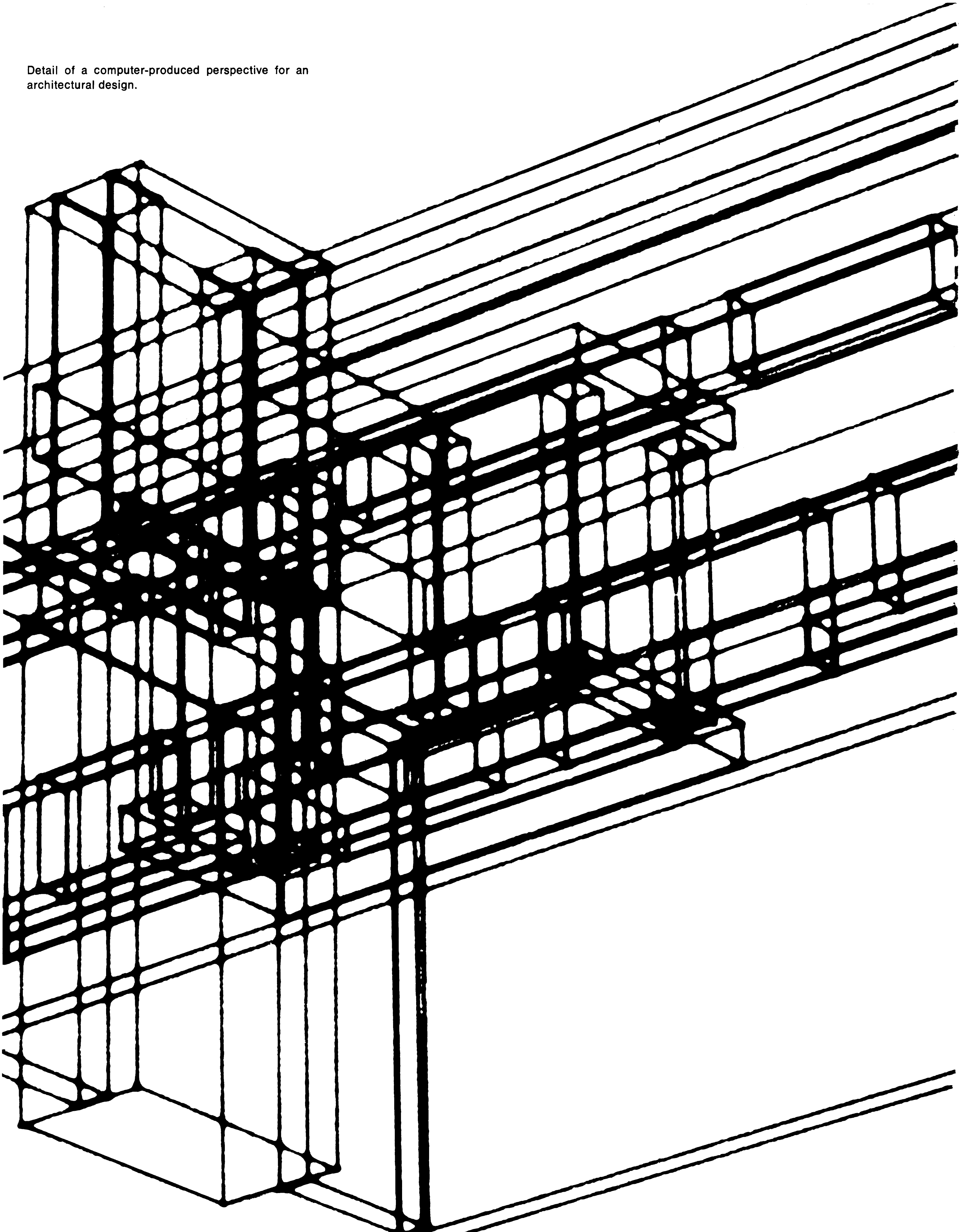
Computer-Aided Design
Author(s): Steven A. Coons
Source: *Design Quarterly*, 1966, No. 66/67, Design and the Computer (1966), pp. 6-13
Published by: Walker Art Center
Stable URL: https://www.jstor.org/stable/4047327

Detail of a computer-produced perspective for an architectural design.



Stephen A. Coons is Associate Professor of Mechanical Engineering at the Massachusetts Institute of Technology, where he is in charge of the computer-aided design group. He was previously design engineer for the Chance-Vought Aircraft Division of the United Aircraft Corporation where he developed mathematical methods for describing the shape of aircraft fuselages by computer. Professor Coons is author of numerous papers and articles on design, the geometry and description of shapes, and computer graphics.

In the following article, Professor Coons analyzes the human and mechanical aspects of the creative process and proposes to let the machine, in this case the computer, take over where the task becomes repetitious and non-creative. In the description of Sketchpad, an early man-machine graphical communication system, Professor Coons demonstrates how drawing is used as a means of communication with a computer for the purpose of design and engineering. This system contains input, output and computation programs that enable it to interpret images directly drawn on a computer display. For instance, it is possible to draw a basic bridge shape onto the display tube, a television-like display, and then experiment with various loading conditions and different supports. The computer produces a visual display showing the changes resulting from the application of different loads and the effects of minor modifications of the bridge design.

In a further development of this system it is possible not only to work with two-dimensional but also three-dimensional images, which, of course, opens this system to the industrial designer, architect and graphic designer.

COMPUTER-AIDED DESIGN

by Steven A. Coons

HUMAN AND MECHANICAL ASPECTS OF THE CREATIVE PROCESS

While designers in engineering, perhaps, are less interested in aesthetics than designers in other fields, all creative designers are involved in a similar process. This design process unfolds something like this: at the beginning, in the design of a device or system (be it a motion picture projector, an airplane, an automobile, or a battleship), the designer does not have a very clear notion of what he wants to do. He has only a vague concept, or none at all, of how he will go about accomplishing his task. In this sense, the design process is a learning process during which the designer must learn what the problem is and how to solve it. Within this process of learning there are certain exciting aspects of discovery. But these are interspersed with long tedious periods of rote behavior — sheer unadulterated dull work — noncreative but necessary. It is appropriate to have computers to do this noncreative work so as to leave the designer free for the activities human beings are good at: innovation — the association of hitherto unrelated ideas. The typically human aspect of the design process is invention: the grasping of schemes that are at the beginning vague, tenuous, dream-like, and solidifying them into something tangible that can be looked at, explored qualitatively, and evaluated quantitatively. To the same schemes, one can apply analytical procedures and then, on the basis of these procedures, make more precise judgments. While all activities during the design process up to the application of analytical procedures are humanoid, analytical procedures are essentially not.

As another example: mathematics, however much a man may enjoy it, is to a large extent a mechanical and not, strictly speaking, a human process. Human beings cannot follow through step by step every detail of a mathematical proof. They cannot multiply two ten digit numbers together without, in each case, a piece of paper and a pencil in order to temporarily store some of the details involved and think up some more details to write down. On the other hand, the strategy by which we prove a mathematical problem can be dealt with in a man's head without pencil, paper or other artificial mechanical means of storage. There is a dichotomy between what people can do with ease and what is mechanical and therefore should be done by machine. The details of a mathematical exercise, whether it be a proof or an execution of a computation, are usually too cumbersome and complex for human creatures to remember and should be done by machine, while the construction, or the structuring of the strategy that yields the mathematical exercise, is a process that people can and should do. Mathematics, as design, is an example of a creative process.

While engineering designers have often less to do with aesthetics and more with mathematics, even mathematicians have some sense of the aesthetic in their proofs. Actually, designers in engineering, architectural designers, industrial and graphic designers, whether they pay a great deal of attention to aesthetics or not — even mathematicians and physicists — all of these people are involved in a similar process in that they, at the beginning, structure ideas, form concepts, produce associations, examine tentative trials qualitatively, behave as generalists of ideas and then subsequently test these ideas by various techniques — mathematical, computational, mechanical.

REQUIREMENTS FOR MAN-MACHINE INTERACTION

In order to make a computer an assistant in the design process and in order to make it do the part of the design process that is noncreative work, several requirements must be met. The problem is to make it possible for a designer and a machine to work on problems together — the designer doing what the machine can't do and the machine doing what, in a sense, the designer can't do (like evaluating the product of two ten digit decimal numbers). One important requirement is that the designer can talk to the machine in a natural way, using natural forms of communication. Natural forms of communication with a computer, or better still, natural ways of communicating ideas and information include, of course, the graphical form.

Another important requirement is that the designer can interact with the computer as though the computer was paying strict attention only to him. But this must be economically feasible because computers and their services are very expensive. As a rough but conservative estimate, it costs $10 a minute to use the computer. In the creative process, the designer may sit vis-a-vis a computer and while away time thinking. It would be a waste of money to have the computer wait patiently for the designer to think up a new idea. Consequently, as of recent years, a new concept of time-sharing a computer has been developed. This means that the designer can sit with the computer, "holding its hand," so to speak, and talk to it, while the computer will seem to pay attention only to him. But, in fact, the computer pays attention to the designer for a very short period of time and then turns away from him and looks at some other user, pays attention to him, and then turns away from him and pays attention to someone else, and so on. The computer may pay attention to as many as thirty, forty or one hundred different people who are talking to it, communicating with it, having it work on their problems. It executes or takes appropriate action in response to their commands, wishes, and their communications so rapidly that each individual user believes that he has the undivided attention of the machine. In this way the computer is kept constantly busy and all the users of this system are busy and happy and don't have to feel that they are wasting money if they do wish to sit and think.

To sum up the idea of man-machine interaction: the designer sits at some kind of a terminal device — a console — connected to a computer system. There are many such consoles with many users seated at them working on design or other problems and sharing computer time. The interaction takes place through all forms of communication such as graphics, mathematical symbolic statements or ordinary English.

Dr. Ivan E. Sutherland working on a Sketchpad display tube mounted on the console of M.I.T.'s Lincoln Laboratory TX-2 computer. Push buttons used to control specific drawing functions are on the box in front of him. Size and position of the part of the picture seen on the display tube is regulated by means of the four black knobs below the screen.

THE ORIGINAL SKETCHPAD SYSTEM

An important first step toward *graphical* communication between the designer and the computer was Ivan Sutherland's "Sketchpad" program written for the TX-2 computer of the Massachusetts Institute of Technology, Lincoln Laboratory. This program was completed in 1962. In computer technology, something that is four years old becomes very quickly worthless, including computers themselves. In this sense a four-year-old computer item is certainly an "antique". From this point of view the original Sketchpad system appears almost as a remnant of the past.

But, nonetheless, this system could do some things which were very sophisticated. It had a console so complex and confusing that it could startle and frighten any operator: toggle switches activating programs for "erase," "start drawing," etc.; dials; blinking lights; a light pen; a 7 x 7″ cathode ray tube, etc. — but this is not too important. An early radio, too, had numerous knobs and was completely open so that the tubes were exposed. In order to receive a station, all the knobs had to be adjusted in a kind of trial and error fashion and there was no clear, descriptive notion about what to do in order to tune in a station. All the dials were, in a sense, cross-coupled. The original Sketchpad system bears the same relationship to a modern console as an old fashioned radio with all its knobs to a modern television set.

Sketchpad was purely graphical and purely geometrical. One could draw with the light pen on the screen — straight lines, circles and other surfaces. It could not solve any problems that had to do with abstraction, other than the abstractions of geometry itself. The operator could impose constraining relationships such as: "Make these two lines parallel" to the computer. That, of course, is a geometrical abstraction and the computer program could follow such an instruction. But the designer could not say to the early Sketchpad: "This line represents a piece of structure with a certain thickness and with certain cross-sectional characteristics, made of a certain material and obeys certain physical laws."

Sketchpad had toggle switches that commanded the computer to satisfy such constraining relationships as the operator imposed. There were four knobs beneath the screen of a cathode ray tube which were used for four kinds of motion applied to the drawing on the screen: a rotation, a horizontal translation, a vertical translation and a change in magnitude. The precision of the graphic information presented on the screen is one part in ten million which means one thousandth of an inch in eight hundred feet! This made it possible to enlarge, look at a small region, change it and then push it all away from the viewer (by reducing its size) so that the entire structure could be visualized.

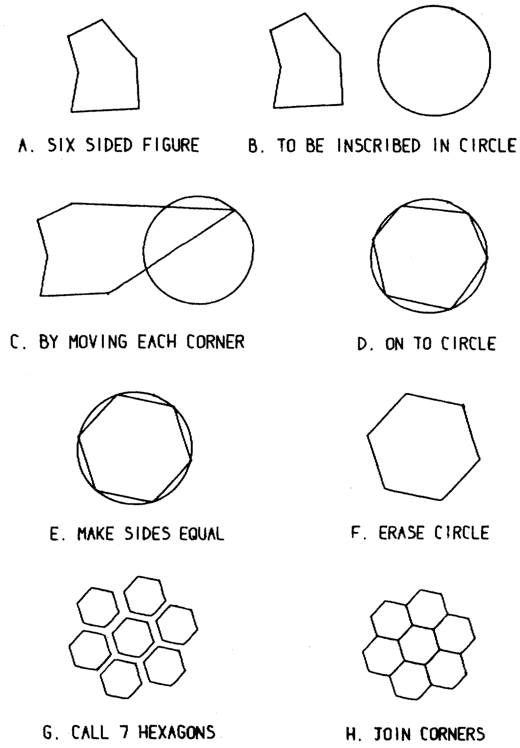
Sketchpad could attach one line to the end of another separated line by means of toggle switch instruction, even if the light pen would not attach the lines precisely on the screen. The computer interpreted the instruction in such a way that the two lines were attached with mathematical precision at the end point, so that they were truly concurrent.

The foregoing example shows that the computer can interpret meaning. The interaction between operator and computer goes somewhat like this: the computer "says" to itself: "I think what the boss means is that these two lines should be concurrent." Therefore, subsequently, if "the boss" pulls the figure apart, the computer will put it back together again. Now the computer has many ways of reassembling the figure. It does not know exactly in what way "the boss" wants the lines attached, it only knows that it has been told to attach them. So it will choose, automatically, one specific way of attachment and if that turns out to be appropriate — fine. If the operator does not like the decision the computer makes, he can talk with the computer and say: "No, I didn't mean that, I meant something else." In other words, it is not necessary nor even desirable in a man-machine interactive relationship that the computer be taught to consider all contingencies and pick the best one. It is only necessary that the computer make some response. Then the operator can interact with the computer and modify the response in a direction that is appropriate to his purpose. This is very much like the psychological process called "reinforcement." In this case the behavior of a machine instead of an organism is reinforced.

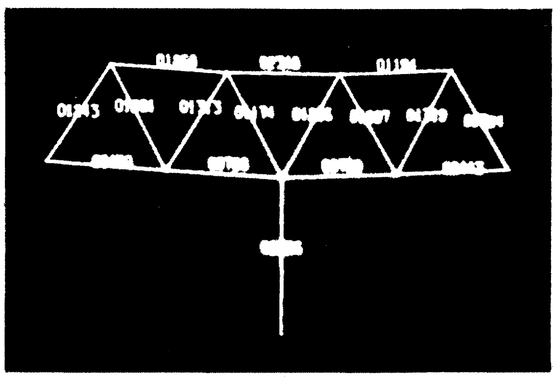
To the extent that a design problem was of geometric nature and did not involve problems requiring expressions in non-geometrical terms, Sketchpad could solve design problems. The operator could, for example, draw a cross-sectional picture of a little angle bracket on the cathode ray tube with the light pen and then "drill a hole in it" by adding the appropriate lines to indicate the hole. Then he could draw separately a rivet, play with the shape of the rivet until it suited him, call for the bracket and for the rivet at the same time and then put both together.

ARCHETYPES AND ICONS

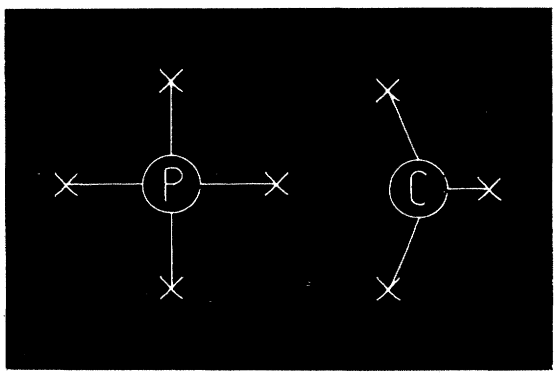
Although with the original Sketchpad only straight lines and circles could be drawn, it



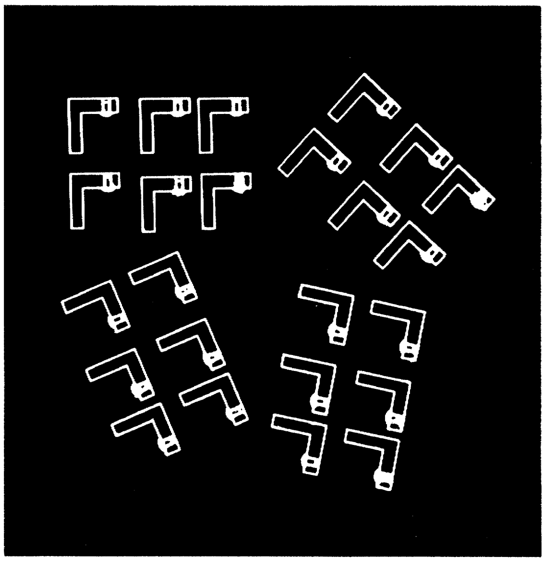
A pattern of hexagons produced with Sketchpad. The designer points the lightpen at the display and presses a button called "draw" and the computer will construct a straight line stretching like a rubber band from the initial to the final location of the lightpen. Repeated pressing of the button will produce additional lines forming, for example, an irregular six-sided figure. In order to make this figure a regular hexagon, the designer can inscribe it in a circle. The circle is drawn by pointing the lightpen at the screen and pressing the button "circle center." This leaves a center point on the screen. Choosing a radius, pressing the button "draw" again and moving the pen in an approximate circle, the designer can cause the computer to construct a perfect circle with the chosen radius. Using similar operations, the corners of the six-sided figure are moved one by one into the circle, its sides are made equal, the circle is erased, the number of hexagons is automatically increased and finally joined at the corners.



Display from Sketchpad System showing stresses resulting from application of load via lightpen to a geometrical representation of a bridge truss.



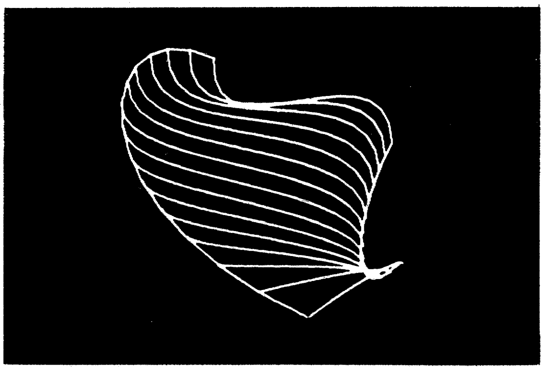
An "icon" or display of a constraint such as "parallelism" (P) or "circle" (C), can be called forth on the screen to tell the operator that the computer has applied a constraint.



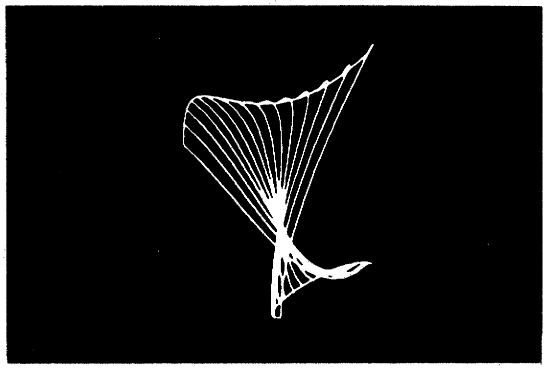
Compounding of arrays of similar objects. Once the fragment is designed it becomes an "archetype" which, if stored in the computer, can be replicated any time. The display shows four arrays each consisting of six identical elements to make a compound array of 24 fragments.

embodied several very important ideas. One was the principle of "archetypes." If one were to draw a rectangle, using straight lines, then it would be never again necessary to construct a rectangle, because the simple rectangle, once drawn, becomes the generic archetype of all rectangle-ness for future use. Likewise, if one designed a "thing" (such as a rivet), and that "thing" was important enough to be used again and again, it would become in some sense an archetype and one would never have to design it again. It then would exist in the computer and could be replicated at any time.

Another important idea used in the original Sketchpad system was the notion of constrained relationships. In any random sample of two lines drawn haphazardly (as if one were to throw thin sticks on a surface), most pairs of lines will be intersecting. If the operator wanted to make them both parallel and equal in length, he could communicate his wishes to the computer by calling for two atomic constraining relationships. The first was that the lines be parallel, and the second, that they be made equal in length. When the operator did call for these constraining relationships he certainly could remember whether he had or had not called for these two constraints. But not all situations might be that simple — indeed the operator may wish to impose many other atomic constraining relationships. Therefore, since there is a potentially very complex situation, the computer should be able to exhibit an "icon," a graphic symbol, which stands for the atomic constraining relationship. The icon should be a sign which tells the operator that the computer has thought of and applied a constraint. At the same time it should also communicate to the computer that the constraint exists. The icon should have transparency — allowing the computer to look out at the operator and see his wishes and allowing the operator to look into the computer and see that the computer has paid attention.



Two views of a free-form surface as generated by the computer. The design information consists of the four boundary curves; the computer supplies the smooth internal contours in about one tenth of a second. The surface can be modified in any way the designer sees fit.



A free-form surface rotated in space to reveal its shape more clearly to the designer.

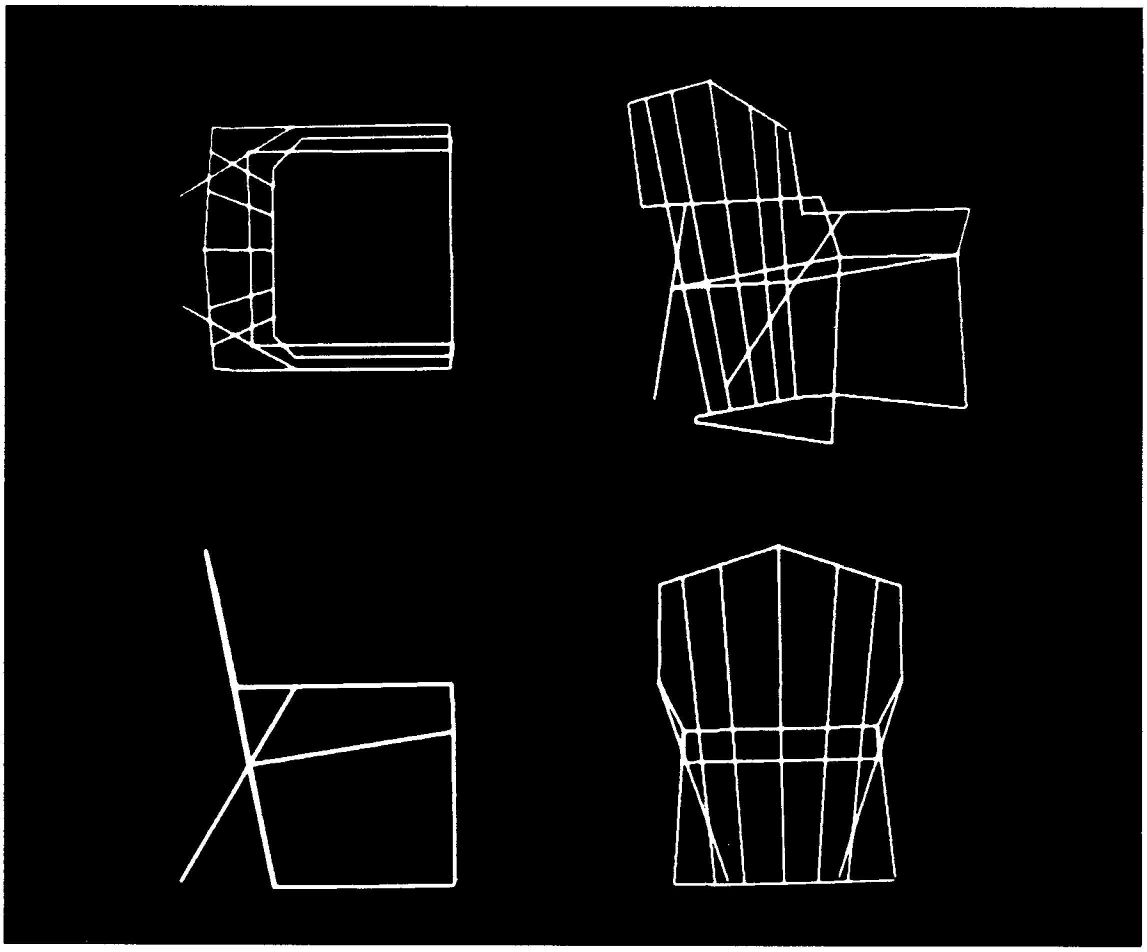
The original Sketchpad had such an icon for making lines parallel and another icon for making lines equal in length. There were other icons for making lines horizontal or vertical when they were drawn nearly horizontally and vertically by the light pen. There were nineteen such constraining relationships capable of being called for by the operator. In today's system, there are a great many other constraints and relationships that can be invoked, but these nineteen served in the past when Sketchpad was written. Although the notion of making a graphic icon to represent constraints is not done today in the same way Ivan Sutherland thought about it, icons are still used, and constraints are applied to the objects or the elements of pictures produced in such systems. In this sense it is very appropriate to credit Mr. Sutherland with the scheme.

Timothy Johnson working on a Sketchpad 3 display.

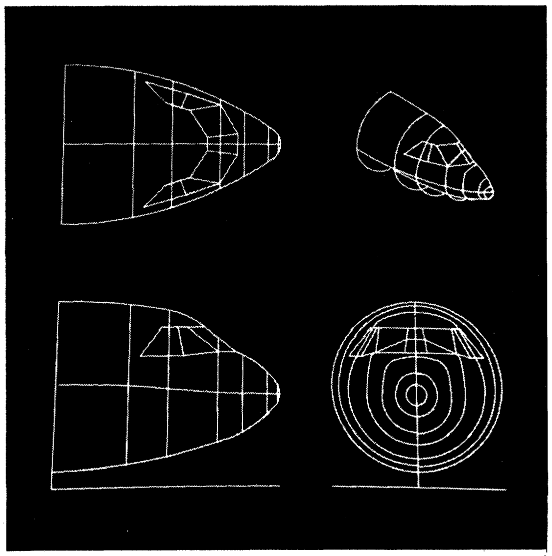
SKETCHPAD 3

The classic Sketchpad system was followed by "Sketchpad 3." This name indicates an extension of the two-dimensional Sketchpad into three dimensions. Sketchpad 3 allows direct communication with the computer, which results in three-dimensional images. For instance, the operator can see a three-dimensional surface, and furthermore, have it rotate in space. In one sense, Sketchpad 3 is less sophisticated than Sutherland's original version which was a new idea. But in another sense it is more sophisticated. After all, it is easy to draw two-dimensionally, but it is not so easy to teach the computer that there is a three-dimensional space and that the cathode ray tube is not the world or the universe, but that there is something beyond it.

Sketchpad 3 was developed by Timothy Johnson, who, at M.I.T., used the basics of Sutherland's Sketchpad and built on them. In his system the face of the cathode ray tube is divided into four quadrants: one quadrant is for the plan, one for the front view, one for the side view and one for the perspective view of any object one might want to draw. If the plan and the two elevations are drawn, the perspective view of the object will "automatically" appear in its quadrant. Likewise, the conse-



Sketchpad 3 display showing three orthogonal projections and a perspective view of a wire-frame chair.



Display from the System Science Section at the Lockheed-Georgia Company, Marietta, Georgia, showing top, side, front and perspective views of the forward fuselage of a large transport aircraft.

quences of any change the designer makes with his light pen in any one of the four quadrants are "automatically" shown in the other three quadrants.

Sketchpad 3 allows for rather complex visualizations. For example, a mosaic-like pattern wrapped around the surface of a sphere would be very tedious to construct with pencil and paper. To do it with Sketchpad 3, the designer would use his light pen and switches as follows: he would start with a vector and a fixed point and establish a drawing plane on the right hand end of that vector. The vector becomes a way for visualizing an invisible sphere. Then he would draw diamond shapes. These would be attached (by pulling them into place with the light pen) to that invisible sphere. If the designer were to decide on a smaller diamond, he could erase the original and then draw one in the desired size. Next he would rotate the front view about an axis to put the invisible sphere into a new position. Now with the sphere in a new position he would draw another diamond. Then the sphere is rotated again. At this point he could stop drawing diamonds because the computer now knows how to go about drawing diamonds and the designer has an inexhaustible supply of prototype diamonds and a way of attaching them. Two rotations have taken place and the sphere can now be put in any general position in space.

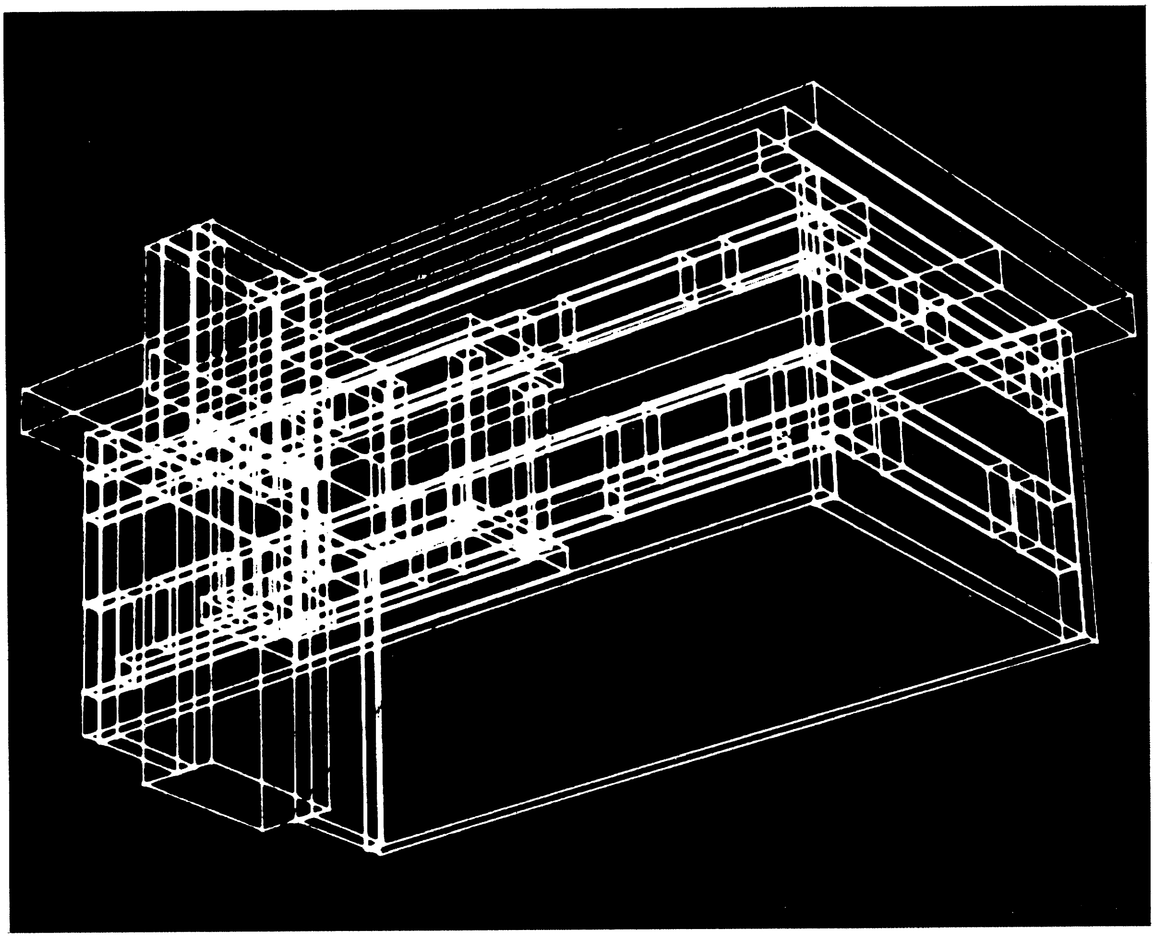
The computer does not know that a sphere happens to be a relatively simple surface. It turns out that the mathematics or the representational structure inside the computer does not distinguish between the complexity of that particular surface and that of a battleship, an airplane, or a tobacco pipe. Today one can draw virtually any kind of a shape: the hull of a ship or even freehand sketched curves. It will not be long before the designer can literally do sculpture with a computer. He will be able to draw three-dimensional configurations as complex as the human figure. The computer will then have an internal structuring of information so precise and so detailed that it will be able to probe any independent point on the surface of the three-dimensional human figure that exists only in the computer's data structure, and to determine its surface with great precision.

NON-GEOMETRIC ABSTRACTIONS

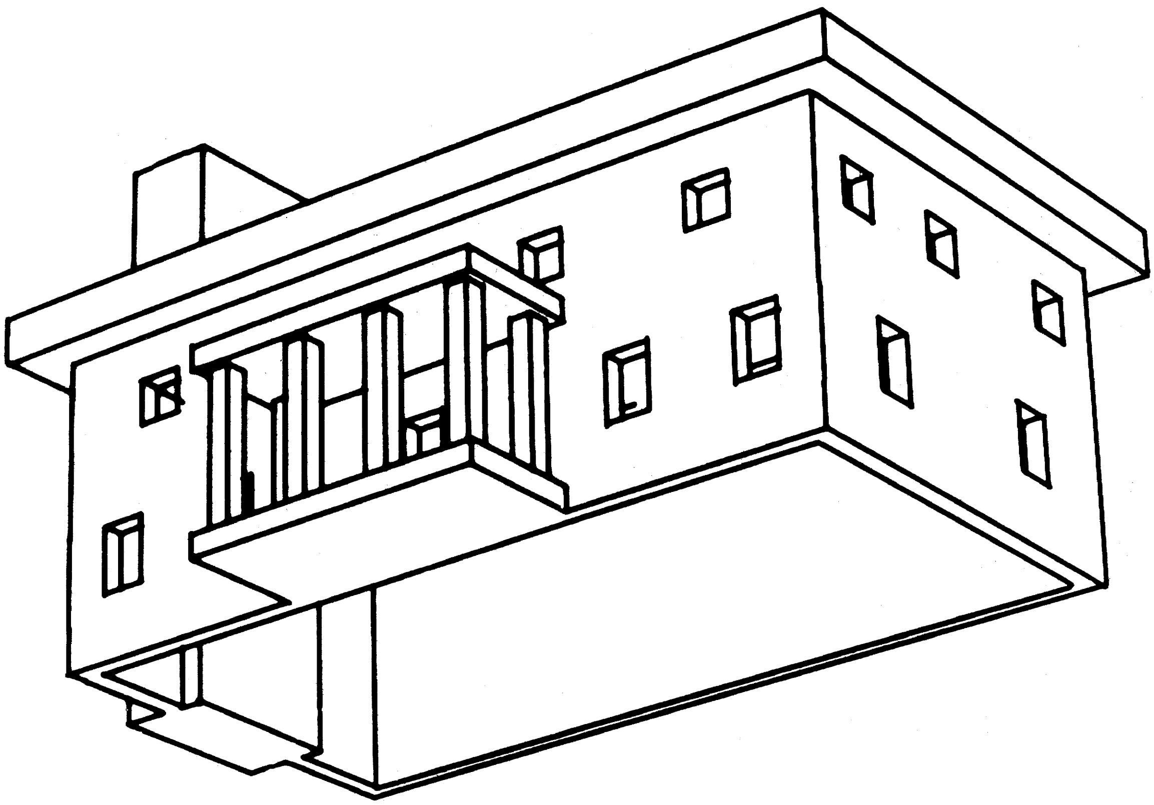
With the original Sketchpad system it was not possible to solve any non-geometrical problems. Today's complex systems, however, are able to solve problems containing other than geometrical abstractions. The designer can draw a diagram which does not represent a "thing" but an "idea." This idea might be that there is a number and another number and a process adding these two numbers together. This may be drawn with the light pen on the cathode ray tube in the form of a diagram. The computer's output, the result of that drawing procedure, could be stored or some other mathematical operation could be applied to it. The diagram would show procedures, inputs and outputs of procedures, and tie all this together. This amounts to putting numbers in one end of this diagram, causing the computer to interpret the meaning of the diagram, take the numbers, perform the indicated operations on the numbers and yield the result. Such a diagram would not be a graphic representation of an object, but of a totally abstract system.

A designer can also draw a diagram that may be an electric circuit of any degree of complexity. He then can describe to the computer what each of the elements of this electric circuit means. Having done that, he can find out what the resulting currents and voltages are, across and through various pieces of this diagram. He also can draw a picture of a thing consisting of pure geometry, and then assign to the geometry other kinds of abstractions which are non-geometrical so that geometry and abstractions are mixed together in any desired way but presented visually on the cathode ray tube.

The original Sketchpad system thus introduced the basic tools for computer-aided design and established some of the principles of a man-machine interaction. Sketchpad 3, with its development of a three-dimensional presentation of images, surfaces and shapes, increased the possibility of applying this potential to many more areas of design and engineering. Today, with much more complex and efficient computer systems available, the designer is not only involved in the innovative function of design and the mathematical process of analysis, but by transmission of the final design solution directly to the tooling process he can also affect the final manufacturing of the product in question.



A perspective view of an architectural design. A program can be prepared that will cause the computer to display every part of an object. In the illustration below, lines that would not normally be visible have been removed.



It is difficult to write a program that will cause the computer to remove the hidden lines from the display. Even with an appropriate program it takes the computer a relatively long time to carry out the instructions. The program for this display was written by Lawrence G. Roberts of the Lincoln Laboratory at M.I.T.