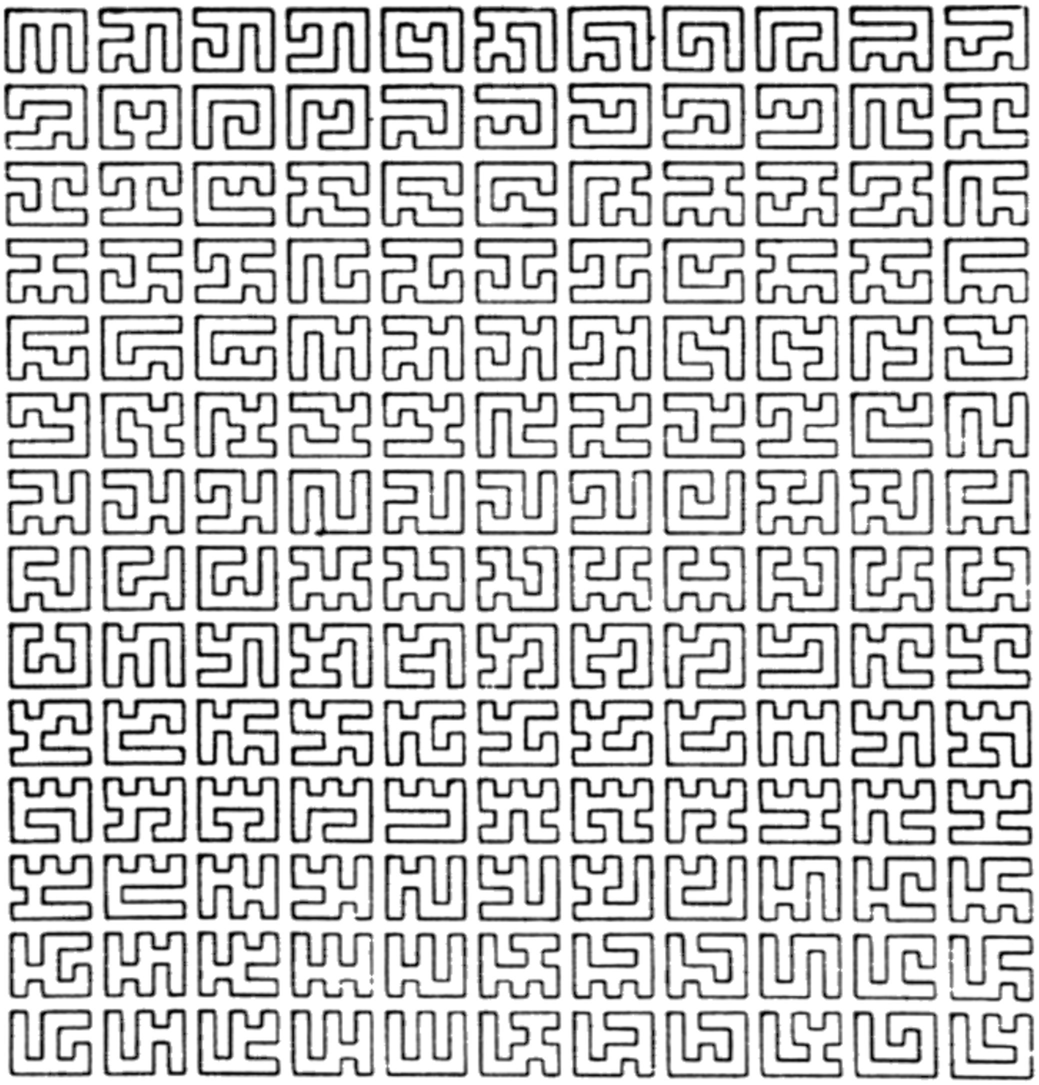
DESIGNING LETTER-LIKE SHAPES

The appearance of a text, written in an unfamiliar alphabet, will almost always be aesthetically pleasing to me. This fact has intrigued me for some time, so I have tried to find an explanation for it. What I believe to be the essential thing, is that the characters have shapes which are so different and yet, at the same time, so strikingly similar. This theory can be tested, of course, by designing sets of forms that are different but similar. The difficulty is, however, to operationalize the notion of "similarity" of forms. Moreover, not each set of similar forms looks "alphabet-like", but, apart from the requirement that the forms can be regarded in principle as consisting of strokes, I have no idea what plays a role here. I hope once to be able to program a computer to design character sets where each new set will come as a surprise - even to the programmer. This is, however, a distant ideal. What is possible now, is to take some (partly unspecified) construction principle and to generate forms according to it.
Some time ago, when I was doodling, I drew a number of letter-like shapes. I observed that these forms had something in common that made them pleasant to look at. So I tried to formulate their common property: may be thought of as consisting of squares; nowhere do four squares clot together; the same holds for the "negative" shape; all parts are connected; there are no holes in it. Although this suffices to define the forms, it is not simple to derive a generating program from this definition. So this problem remained dormant. Then, about a year later, working on an entirely different problem, I hit upon a much more manageable definition:
Consider a rectangle of grid points of a square grid, containing an even number (e.g. 5 × 6) of grid points. Now we have to make a tour of these points, calling once at each point and returning to the starting-point, but in such a way that the total distance covered is minimized.

This problem will have a number of solutions, which correspond exactly to the shapes defined above (and vice versa). Now this does not represent a trivial problem - far from that - but it provides a good basis for a program set-up which is open to many ways of improving the efficiency. An ALGOL 60 program to list all solutions for a given m × n rectangle was written by Michiel Baron, a 17-year old schoolboy. It is one of the most sophisticated



programs I have ever seen. The technique is that of "backtracking", but as the route is built up step by step, each time far-reaching conclusions are drawn by application of rather intricate arguments, showing that certain paths are barred, i.e., cannot be part of a legitimate completion of the partial route. This gives a tremendous reduction of the branching rate and cuts the necessary computing effort to acceptable proportions.
The complete list of solutions for the 5 × 6 case, which is reproduced here, was computed and drawn in 7 minutes time, using the Electrologica X 8 of the Mathematical Centre.

Lambert Meertens