serendipity

Not only computing – also art

JOHN LANSDOWN

Truth is beauty

It will not have escaped the notice of those of you that look at the newest buildings that pattern and ornamentation are coming back into fashion. We see this revival of interest displayed too in the success of the recent republication by Van Nostrand Reinhold of that magnificent book, *The grammar of ornament.* Originally published in 1856, this book has always been a great influence on architects (even masters like Le Corbusier who worked in the unornamented International Style which is now so disliked by popular critics). In its new cheap edition, the book is currently impressing and instructing a new generation of designers.

Its author, Owen Jones, was the architect who acted as Superintendent of Works for the Great Exhibition of 1851 and was in charge of the decoration of the Crystal Palace after it was moved from Hyde Park when the Exhibition closed. He was one of the first designers to have an interest in what we would now call Islamic Art and collected together examples of this from Spain, Persia and other parts of Asia. He was also one of the first to see any significance in the art of what he termed the 'savage tribes' – primitive art which was later to have such an effect on the Cubists and other moderns.

Jones' principles which guided his selection of examples for the book and the way he worked as an architect were set down in a list of 37 'Propositions': a sort of manifesto with pithy phrases such as: '*That which is beautiful is true; that which is true must be beautiful*' and '*The principles discoverable in the works of the past belong to us; not so the results. It is taking the end for the means*'. Although couched in Victorian language and expounded as if they were the ten commandments, his propositions still summarise the views of many designers and give excellent guidance to good practice.

Although he does not deal in any detail with the ways of setting up the patterns, Jones' Proposition 8 is, '*All ornament should be based upon a geometrical construction*'. This leads me to believe that, if he were alive today (he died in 1874), he would certainly have been interested in the way in which computers and mathematics can help create new forms of geometric ornamentation. For example, nowadays we know from group theory that there are only 17 basic ways of tesselating the plane. Had Jones known this it might have further guided his choice of examples from places like the Alhambra which he regarded as 'the very summit of perfection of Moorish art'. Only recently it has been finally shown that all of the 17 patterns are used in the Alhambra (something that, without proof, I have taken for granted since I was a student).

A knowledge of some of the more recent developments in mathematics of tesselations might also have widened the scope of his fascinating book. Few people realise, for instance, that tesselations need not be confined to regular lattice shapes like those in Figure 1 and generally used for historical ornamentation: *any* four-sided shape can be used to tile the plane. To do this you take a quadrilateral and determine the midpoint of each edge. Put down the first tile. All the others are laid by placing matching edges together after rotating the tile 180 degrees about the midpoint as in Figure 2. Two examples to illustrate the point are shown in Figures 3 and 4: they derive from a program that anyone could write. Knowing the principle of rotation about the edge mid-point, it's a short step to realising that, providing the shape of a particular edge is 180 degree-symmetrical about its midpoint, countless different forms of Escher-like interlocking tiles can be created (Figure 5).

I've recently been looking at more general forms of interlocking tiles and have developed a computer-based ways of devising shapes of arbitrary complexity for these. Figures 6 and 7 show some of the simpler ones.

But all these examples are *periodic.* That is to say, the patterns they create are regular: one part of the design is exactly the same as another and can be derived from it by a simple process of

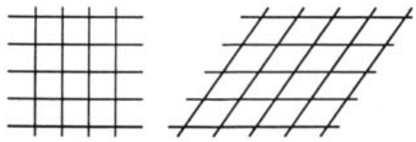

Figure 1

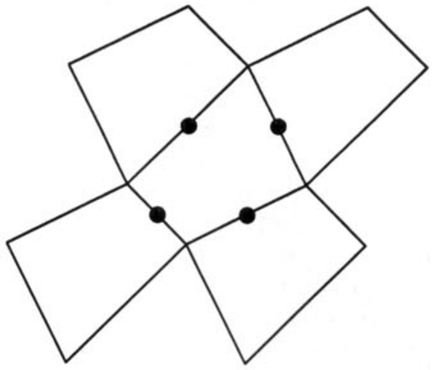

Figure 2

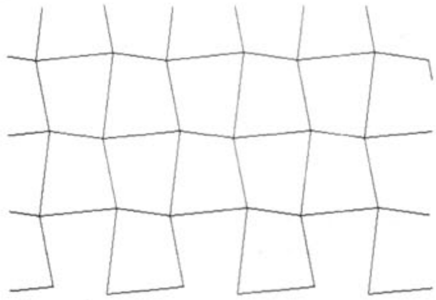

Figure 3

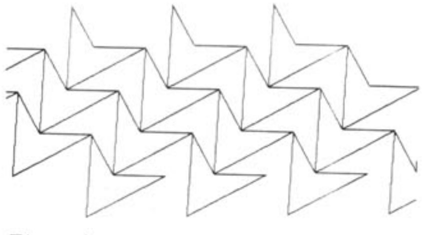

Figure 4

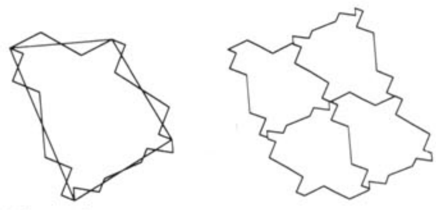

Figure 5

translation. Surprisingly, though, it is also possible to tile the plane in a *non-periodic* way. Mathematicians have known this for some time but once thought that it was not possible to devise tiles which could *only* be used for non-periodic tiling. However, as pointed out in a new paper by Rangel-Mondragon and Abas (shortly to be published in *Computer graphics forum*), in the early '60s a mathematician showed that there was indeed a set of tiles which would cover the plane non-periodically but could not do so periodically – unfortunately more than 20,000 different tiles were needed for this to happen. In the early '70s, Roger Penrose showed that, by using only two four-sided tiles, the plane could be covered in a non-periodic way and that these two tiles could not work periodically.

The way of writing a computer program to plot these tiles is given both in McGregor and Watt's excellent book, *The art of microcomputer graphics*, (Addison-Wesley 1984) and in the Rangel-Mondragon and Abas paper. Figure 8 shows a layout based on their methods.

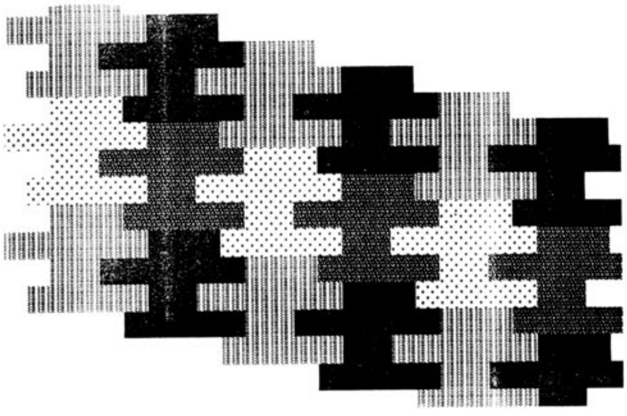

Figure 6

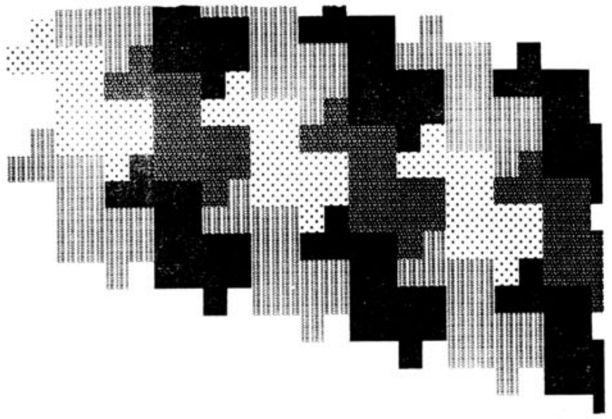

Figure 7

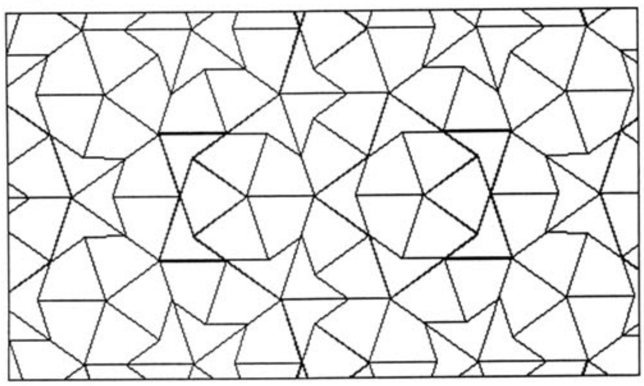

Figure 8

If they asked me I could write a book

About ten years ago I drew readers' attentions to a book on Basic by Donald Alcock. Apart from being an excellent introduction to that popular language, the book was unusual in being lettered and illustrated entirely by hand. Since that time Donald has produced a number of computer books in this unusual style and his latest – and to my mind his best yet – is *Illustrating Pascal*. Once again the free juxtaposition of diagrams and text that his approach allows, together with Donald's clear way of explaining complex ideas come together to produce a work which will interest beginners and experienced programmers alike. He is especially good on data structures and Pascal records and his explanation of Quicksort is a model for textbook writers. I hope he'll tackle C at some time. I'm sure his treatment of pointers and indirection – so much a feature of that powerful language – would be of value to everyone.

I urge you to read this book – especially if you're concerned with writing user manuals. If all these were presented with such clarity, much time and effort would be saved. *Illustrating Pascal* is published by Cambridge University Press.

Tuscan days

In the last issue I promised to tell you about the NATO Advanced Study Institute held in the Summer at Il Giocco, near Pisa. As I expected, it was hard but enjoyable work with lectures starting at 9 am and finishing at 10 pm with breaks only for meals and an afternoon siesta. Many of the younger ones simply couldn't stand the pace!

There was much of interest in the conference and the proceedings will shortly be published by Springer Verlag under the title, '*Theoretical foundations of computer graphics and CAD*'. Virtually all the presentations enlightened me in one way or another but I found the best paper to be one which, because of a family bereavement, the author was unable to deliver. This was a paper on ray tracing by a young American, Roman Kuchkuda.

Using a literate programming style, the paper lists and explains a complete and working ray tracing program (in C). I have put the program on my system and can confirm its correctness (although, if others intend to do the same, I suggest they omit Kuchkuda's accurate but extremely inefficient gamma correction routine: a four times speed-up will result!) The paper is an excellent justification of literate programming – a technique which the BCS has recently endorsed by giving a prize to Harold Thimbleby for his June 1986 paper on the subject in *The Computer Journal*. Malcolm Sabin frequently urges the Computer Art Society to run a competition for computer programs as art works: Kuchkuda's paper would certainly get my vote.