Not only computing – also art

JOHN LANSDOWN

The other Mr Cube

Most people who take an interest in the subject matter of these columns will be aware of an object known as Rubik's Cube. A walk down the High Street of any town reveals them on sale in all sorts of likely and unlikely shops. The street vendors in Oxford Street have trays full of them – including octagonal variations on the original – indeed millions have already been sold all over the world. For those who haven't noticed their existence, Rubik's Cubes are puzzles apparently consisting of 27 small cubes (known to aficionados as 'cubies') solidly connected together in three layers of nine by an ingenious mechanism which allows the layers to be rotated no matter how the cube is orientated. Each face of the cube is differently coloured and, apart from the centre cubie of each face which can only be rotated in place, the outer cubies can be repositioned by rotations in any direction, resulting in a cube of multicoloured faces. Only three or four random moves are needed to completely scramble the faces and the object of the puzzle is to manipulate the scrambled object back into its original form. Without instructions this is an astonishingly difficult task as it is reckoned that there are 4.3 × 10E19 different permutations of the colours!

Now, by any standards, 4.3 × 10E19 is a pretty big number and, even with instructions, the task of resetting the colours takes some doing. However, the puzzle has caught the imaginations of many and competition is on to see who can solve it in the shortest time. About 20 to 30 minutes is needed for beginners – although they have to have the instructions in front of them to achieve these times. Wizards like Bob Parslow take under four minutes; another Computer Arts Society member, Robert Baker (who has a collection of many different sorts) takes less than three minutes; grave old plodders like myself need five or so whilst master puzzlist and inventor of the Life Game, John Conway, is said to be able to do it behind his back – a feat of wretched virtuosity and bravura which sets him beyond the pale! At the time of writing the record is around 40 seconds: a time which has to be compared with computer solutions which take about half a second of CPU time – and computers don't have the mechanical task of manipulating the faces.

Were this just a game it would be of no more than passing interest, but the cube has a much more useful purpose. It was invented in the mid-1970's by Erno Rubik, an architect who teaches at the School for Design at Budapest. He devised it to assist in teaching 3-dimensional design although its main scientific and mathematical interest is that it embodies in a particularly graphic way many of the principles of Group Theory and this aspect is the source of much of the theoretical study of its solutions. A nice introduction to some of the mathematical background is given by Douglas Hofstadter in the March 1981 edition of *Scientific American*, but the main theoretical work is by David Singmaster of the Mathematical Sciences and Computing Department of the South Bank Polytechnic. His privately published little book *Notes on Rubik's Magic Cube* has now reached its fifth edition and is a mine of essential information. Anyone who wishes to go beyond the aching wrist stage to a more fuller understanding must read this book. For my part, I find the cube a useful device for illustrating some of the problems of 3-dimensional computer graphics; in particular, the non-commutative nature of 3-D rotations is immediately seen: a rotation about the Y-axis followed by one about the X-axis is quite different from one about X followed by one about Y.

There is a growing number of books and pamphlets giving solutions to the puzzle. Singmaster's notes has one. A tiny Penguin book using the same notation is *Mastering Rubik's Cube* by Don Taylor. Bantam Books have *The Simple Solution to Rubik's Cube* by James G. Nourse. This uses a different notation but gives quite a good solution. Two other books are *A Simple Approach to the Magic Cube* by Bridget Last (Tarquin Publications, Norfolk) and *Mastering the Rubik Cube Easily* by Czes Kosniowski (Kornworthy Publications, Eastleigh). Those, like me, who appreciate diagrammatical notation more than mathematical will find Last's book quite useful but, for my money, the best of all is *Solving The Cube* by Cyril Ostrup (Beltrans Import and J. J. Neenan, Gothenburg). This gives a diagrammatic solution which, whilst not the fastest, does not require the memorising of a number of exceptions. I use a combination of the Ostrup solution and that of Nourse together with some shortcuts of my own.

For those who want to use their computers to help in the study of algorithms and methods, the August 1981 edition of *Computing Today* gives an 8080 program and flowchart by Iolo Davidson although better could probably be devised. Get yourself a cube and explore its mysteries – at the very least you'll find your understanding of 3-D manipulations improving and you may even astonish your friends.

Drawing the line

Although the advent of full colour raster terminals has moved computer graphics into an entirely new and exciting field where pictorial realism is almost a matter of course, it seems to have had one deleterious effect – that of almost stopping any research and development into line drawing techniques and methods. Researchers seem to believe that, as full colour shaded graphics is now possible, no

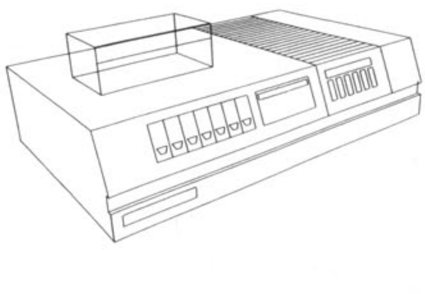

Figure 1

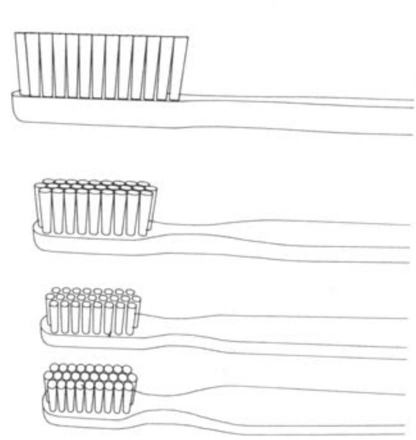

Figure 2

one would want any other sort and much of the new work is devoted to improving this form of output. I often have to prepare drawings, on the other hand, for those who simply want pictures made up of lines.

For reasons which need not concern us, my colleague, Tony Pritchett and I have spent the last few days creating computer drawings of toothbrushes and the special problems of pictorial representation in line have been brought forcibly home to me by this work. Making computer perspectives of hard edged objects built up of flat faces is not too difficult (Figure 1), but real problems arise when you wish to draw objects made up of curved surfaces or having rounded edges – like toothbrushes (Figure 2). When artists make a line drawing of such an object they do so mainly by outlining its silhouette but this is not always sufficient to depict the full 3-dimensional shape. To achieve this, other lines are added, perhaps indicating highlights or changes of tone and shading. Both these tasks are difficult to do by machine without substantial amounts of computation.

As I can't believe that I'm the only one interested in such representations, shouldn't some bright MSc or PhD student be looking at ways of doing them simply and efficiently?

Cruising down the river

Although the Computer Arts Society has had branches overseas for a long time, it has taken nearly 13 years for a new branch to be formed in Britain. Leicester Polytechnic now has a society and I went to what turned out to be its first meeting the other night – and a memorable affair it was. At the invitation of Terry Rowley and his team, we visited the Marconi Radar Systems establishment to see a demonstration of their new ship simulator due to be installed soon at Cardiff (by coincidence, in the Nautical School where I was trained as a boy). This is a truly remarkable system. Set in a full scale mock-up of a ship's b4idge, the simulator presents a wide screen computer produced picture of a seascape continuously changing to take into account the actions of the navigator and pilot, the wind and the weather, as well as the movement of other ships.

The simulated trip we took was from just north of the Isle of Wight into Southampton Docks and we started at night with just the shore and buoy lights visible. As we steamed up Southampton Water at a steady ten knots, dawn broke revealing the shoreline and other ships including a ferry which, due to some pretty poor manoeuvring on the part of our pilot and encouragement from the more irresponsible members of our party, we almost succeeded in ramming. The combination of sights and sounds was incredibly realistic and the whole is a *tour de force* which, I am sure, would be enjoyed by millions if it were set in Disneyland. The system is called Tepigen and, via three video projectors, produces some of the best computer images I have ever seen. What's more, it does them in real time and of recognisably real places. Why doesn't Marconi market the computer graphics system separately from the simulator? Is there anything to beat it?

LETTERS *continued*

Is this not similar to the implementation view of a machine independent operating system?

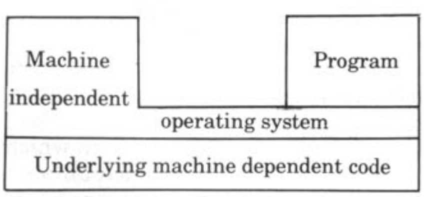

Figure 2

The underlying machine dependent code is necessary to raise the level of the basic machine to some common interface at which the machine independent modules can be executed. With the BCS-JCL model that common interface is defined by the Standard Command Language (SCOL).

In Chapter 3, 'System philosophy' a matrix structure consisting of seven functional areas intersecting five levels of use, is described. It is stated on page 14.

'The computer system which results from this two dimensional philosophy is therefore characterised by:-
a) The entities which are operated upon within it.
b) The operations which may be done using these entities.
c) The responses to the user generated by these operations.'

Surely the entities and operations are entirely dependent on the operating system. It is only the appearance of the entities and operations that are part of the command language or JCL. This underlying dependency of command languages on operating systems means that for machine independence it is necessary to define a machine independent operating system. I would like to see the BCS-JCL model and matrix philosophy continued but with a view to developing an operating system rather than a command language. The command language will follow from the operating system.

Another approach to JCL standardisation is to select a central set of facilities that are provided by all existing commonly used operating systems and to specify a standard set of commands for using these facilities. Three problems immediately arise.
1 What set of facilities?
2 The selection of existing commonly used operating systems.
3 The difficulty of justifying a particular set of commands.

Users requiring facilities outside those covered by the standard set of commands could, *via* a 'drop-through' facility, use the underlying operating system command language. This approach has the following advantages.
a) Providing the set of facilities is small then implementation by means of interpreters is fairly simple.
b) Due to the ease of implementation some degree of standardisation in JCL could be achieved in the near future. Use of the standard could be encouraged by making the interpreters available free of charge, their design having been financed by government grant (*eg* SRC) or by a grant from a professional body (*eg* BCS).

C. J. LAWRENCE *Redhill*

References

1 K. HOPPER, User-orientated Command Language, *British Computer Society Monographs in Informatics*, Heyden and Son Ltd, 1981.