THE MYCOLOGIST GUIDE TO PERSONAL COMPUTING

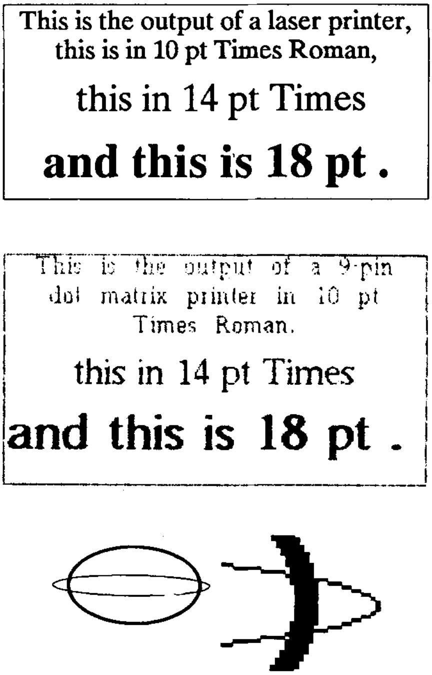
DAVID MOORE
Department of Cell & Structural Biology, Stopford Building, The University, Manchester M13 9PT

My previous article dealt with text manipulation so I will start this one with **desk top publishing (DTP)**. Fully-fledged DTP is an expensive business, requiring a number of major software packages as well as a fair range of equipment. Since DTP programs are rarely better than rudimentary at word-processing and graphics production, high quality programs to suit these needs are usually required in addition to DTP software. Hardware requirements would certainly feature at least one hard disk (30 Mb of disk storage is barely sufficient for serious work) and a printer with 300 × 300 dots per inch (dpi) resolution — either a laser printer or one of the newer diode or crystal shutter devices. The kit might be completed with an electronic scanner; primarily used to 'digitise' drawings for inclusion in DTP documents, but it can also convert typescript into ASCII computer text files using character recognition software. More of a luxury item would be a full page computer display. Most monitors can only display about a quarter of a page at a time, but there are now a range of displays able to show one, or even two, A4 pages. Of course, such screens do not show the page to its final resolution; even one of the best (priced at £2500, just for the monitor!) manages only half the resolution of the printer. Assembling this collection will probably involve spending £2000 or more on software and at least £6000 on hardware.

But do not let this shopping list put you off, it is amazing what can be achieved with simpler and cheaper resources. A number of budget DTP software packages are aimed at 'newsletter' and flysheet production and offer the text manipulation features to suit. As for printing, it is astonishing how good 9-pin dot matrix output can look (Fig. 1).

Rather than text production, though, display work implies work with computer graphics and an important starting point is that there are two ways in which a computer can deal with graphics. One is to build a 'bit-map' of the image — a description of the position and state (i.e. white, black, blue, green, etc.) of every bit which comprises the image. The most accessible 'bits' are the individual **pixels** (= *picture elements*) of the monitor display. The other approach is to record each part of an image as a mathematical description of its geometry. Programs which work through bit-maps are called painting programs, those that work in the latter way are called plotting or drawing programs.

Painting programs are the simplest and so most of the cheaper graphics packages are of this sort. They have two problems to contend with, one is inherent in their approach the other a matter of choice on the part of the programmer. Their inherent problem is that once a bit-map is drawn no further information can be put into it without redrawing. This becomes a problem when you attempt to change the scale of an image. If you magnify it by ten times then the pixels become ten times larger but only the same number are defined for position and state. Hence, a painted image becomes 'blocky' when it is expanded (Fig. 1). In contrast, the resolution of a plotted image is independent of its size. It is stored as a geometrical description, the equation of which can be used to generate the additional information needed to keep the edges smooth as



This is the output of a laser printer, this is in 10 pt Times Roman,

this in 14 pt Times

and this is 18 pt .

This is the output of a 9-pin dot matrix printer in 10 pt Times Roman.

this in 14 pt Times

and this is 18 pt .

Fig. 1: Comparison of output of a laster printer (top) with that from a 9-pin dot matrix (middle). The bottom picture shows the effect of expanding a 'painted' image; the intersection of the two ellipses, which are acceptably smooth-edged in the original (left), becomes very 'blocky' when magnified (right).

the scale is increased. The other problem faced by painting packages has its roots in the fact that the pixels of the monitor display are rectangular, but those of a printer are circular. The consequence of this is that a square on a monitor screen has more pixels in its vertical side than in its horizontal. It will look square on the screen but when printed the image will be rectangular. Thus, the programmer must decide his/her priority: monitor screen or printed page. Most of the simpler packages (and unfortunately some of the not-so-simple) favour the screen image.

Graphics packages have become specialised to suit the demands of the 'business' and 'technical' markets. The latter eventually merge into **CAD (computer aided design)** and **CAM (computer aided manufacture)** where the expression of plotting/drawing programming is brought to a peak. The former tend to feature business graphics and slide shows, the emphasis being on high quality presentation and effective communication of information. Computer graphics 'slide shows' may comprise conventional projection slides (the computer image being converted to a photographic one) or may be stored on computer disk together with a 'run-time' version of the graphics program for display on other computers which lack the originating software. This is the basis for computer-displayed sales presentations and those tutorials which come with some machines and software.

Since the computer can be made to create and display a sequence of images, it could clearly display them at such a rate as to create moving images. The limitation is memory storage since keeping a record of every pixel on screen demands a lot of memory even for a single image, let alone for 20-30 per second of 'film'. However, it can be done and will undoubtedly become more accessible as optical ('compact disk') storage devices become available as reasonably priced peripherals for personal computers. There is some hope for budding PC-Spielbergs, though, in the shape of animation programs. These are drawing programs which have the additional facility of 'tweening' — calculating undrawn frames *between* two hand-drawn graphics. Mention of specific items has been avoided in this series but here I have to identify one program which is outstandingly impressive — it is *Fantavision*, published by Broderund Software Inc. for the Apple II computer. This program allows you to draw a beginning frame and an end frame and will then show how the first can be converted to the second in the form of an animated cartoon. I hope that this sort of program can be used to 'show' intermediate stages in mushroom development. At least that's my excuse — it's actually great fun to work (play?) with!