

8

Graphics in DTP

In this chapter we shall be studying the non-textual part of DTP. By this I mean not only the picture elements, those parts that are obviously graphics because they have been drawn, painted or otherwise derived from some form of image, but also the graphical components of the DTP software itself. We shall see what sources of illustrations there are and what can be done with them once they are part of the page on screen.

Lines and Arrows

Immediately above these words you can see the simplest example of the graphics that DTP can provide with its own facilities: the rule-off. DTP's drawing power ranges from the simple horizontal or vertical line, through straight lines drawn *ad lib*, to straightforward shapes such as circles and rectangles.

There are two principles at work here: one is where you include one or more rules as part of a style definition (as in *Impression*) so that it relates to a piece of text, and the other is where the line is a separate graphical element in its own right (as in *Acorn DTP* and *Ovation*) so each one is independent of the text structure. To see the difference let us look at *Impression* first, as its line-drawing capabilities are more basic and closer to the textual aspects of the previous chapter, then at *Ovation*, the next in line, and finally at *Acorn DTP*, whose drawing tools are reasonably comprehensive.

Lines in Style Definitions

The type of horizontal line (*rule-off*) considered here, which is unique to *Impression*, is different from an underline: whereas underlines appear only under specific words or letters and not elsewhere, the rule-off is placed under (or above) the whole of a paragraph and extends between specific left and right positions. It needn't run from

one edge of a frame to another (though versions of *Impression* earlier than 2.09 used only the full text width), which gives you plenty of flexibility over positioning.

Both horizontal and vertical rules in *Impression* are positioned along a style ruler in the same way as margins and tabs; Figure 8.1 shows the ruler from version 2.09 (compare it with Figure 7.5 from earlier versions, which was used there because it was less crowded). The two new symbols before the tab stops are the left and right extents of the horizontal rule-off for the style; like the left and right margins, you can have only one of each. The symbol at the right is for placing vertical lines, and it is treated exactly like a tab stop. You will realise that this means that such vertical lines cannot be placed outside the text frame.

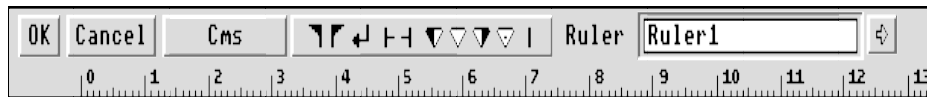



Figure 8.1 The style ruler from *Impression* 2.09

In a necessary division of labour the ‘Edit style’ dialogue box, rather than the ruler itself, governs the vertical positioning of the horizontal rule and also controls the thickness of both horizontal and vertical rules. This is where the ‘Style lines’ area in Figure 7.7 comes in. The ‘Rule-off colour’ button specifies the colour of vertical as well as horizontal rules. Because you can only apply thickness and colour to these rules from the style dialogue box, you might think that if you define rule positions only on a ruler rather than as part of a style they won’t show up; however, if you edit the ruler under ‘Edit style’ it becomes a style and you can give the rule a thickness.

After a bit of careful setting up of these rule controls in both directions it is very easy to put together ruled tables and grids. The verticals by themselves can be used to emphasise text by drawing a line alongside the important passages (or you can use it to draw attention to places where the wording of a document has been changed), as long as there is a margin inside the text frame for the rule to be drawn in.

Lines and Arrows

Ovation’s line-drawing facilities allow you to place straight lines and arrows anywhere on a page, and at any angle. The line tool, , is used for this purpose. Once drawn, a line can be altered comprehensively both in position and attributes; the latter are controlled by the ‘Modify line’ dialogue box shown in Figure 8.2. An arrow is simply a line with a point at one or both ends.

By using the ‘Duplicate line’ option in the Object submenu you can also make copies of a line (or, if a frame is selected, a frame) across, down or diagonally on a

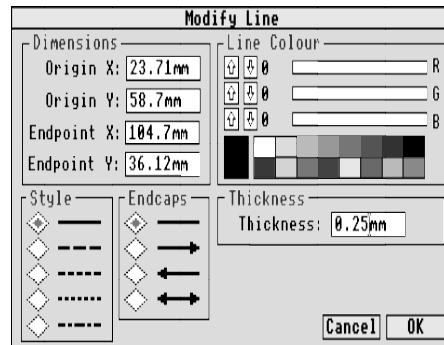
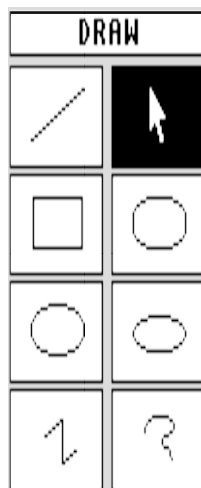


Figure 8.2 The 'Modify line' dialog box from Ovation

page. Again, this is useful for grids or tables (or, by duplicating frames, you can quickly draw up a crossword blank; altering the background colour in some frames will give you the black squares), but note that the lines are in no way tied to the text: any text you wish to be associated with the lines has to be aligned independently. Of course, you would be more likely to enter the text first and fit the lines in afterwards.

Lines, Squares and Circles

We turn now to *Acorn DTP*, which has the most extensive range of drawing tools. In fact they can be looked on as a subset of the tools available in *Draw*, and they work in much the same way. After entering graphics mode by clicking on the pencil icon in the browser (see Figure 6.10), the pointer becomes a large arrow and the display in the browser changes to the eight icons shown in Figure 8.3. The functions of the icons are, in order: straight line, editing tool, rectangle, rounded rectangle, circle, ellipse, 'polyline' (multiple straight lines) and freehand line. The use of the mouse buttons for the last two is rather non-standard but is easily picked up.



With these devices you can easily prepare simple graphics as part of your DTP document, and as in *Ovation* the position and the attributes (thickness, line style, arrowheads) of lines, and the fill tints of shapes, can easily be altered after drawing.

What for? This question must be asked, however: why should you want these facilities when, thanks to the multitasking properties of RISC OS, you can use a genuine drawing package such as *Draw* and simply import the desired item straight into

Figure 8.3 The browser icons in Acorn DTP's graphics mode

the DTP application? There are two possible answers to this: shortage of memory, and convenience. The former is probably only of relevance to *Acorn DTP* running in 1Mb of memory, and the latter is a matter of opinion.

Graphics for Importing

The rest of this chapter is concerned with what you no doubt think of as ‘real’ graphics: sprites and *Draw* files. Before we look at how DTP software can handle them, let us see where they can come from, starting with primary sources.

Software for Originating Graphics

Pictures of one form or another can be prepared on a variety of software, starting with *Paint* (for those with a great deal of patience) and *Draw*; there is no need to go into their basic operations here.

A sort of big brother to *Draw* is a piece of ‘shareware’ available at low cost from Archimedes public domain (PD) libraries: it is called *Draw+* (on Shareware 34 from Norwich Computer Services) and contains several enhancements over the original Acorn product. For instance, there are key shortcuts to sidestep the menu-only operation, and new facilities such as polygon creation, skew manipulation and text to path conversion (like *FontDraw*). One thing it won’t do that *Draw* does, however, is import files in the so-called DXF format, but this is unlikely to worry many users.

Of course, any programs capable of producing sprites or *Draw* files are potential sources of illustrations. Once you get into DTP you can find yourself perpetually seeking new experiences – the temptation to become a DTP graphics junkie is strong. Here is a non-exhaustive list of graphics software, in no particular order and with no recommendations implied (see Appendix 2 for addresses of suppliers):

- **art** – *Revelation* (Longman Logotron), *Artisan 2* (Clares), *Pro-Artisan* (Clares), *Atelier* (Minerva), *Prism* (XOB)
- **3D drawing** – *Euclid* (Ace Computing)
- **ray tracers** – *Render Bender* (Clares), *SolidsRender* (Silicon Vision), *Quick Ray Tracer (QRT)* (public domain)
- **computer-aided draughting (CAD)** – *SolidCAD* (Silicon Vision), *LinCAD* (Lingenuity), *WorraCAD* (Oak Solutions), *Designer* (TechSoft UK)
- **presentation graphics** – *Hotlink Presenter* (Lingenuity), *Schema* (Clares), *GraphBox* (Minerva)
- **miscellaneous** – *ShowPage* (Computer Concepts)

If you don't already have any of these, but like the sound of them, you should read the specifications and any reviews you can lay your hands on and check whether they are likely to meet your requirements. Better still, try before you buy (always a good idea if you can), and shop around to get the best prices.

When you are buying a computer painting package it is worth remembering that those with the most comprehensive options are not necessarily the best for your purposes. If you want to produce representational art, avoid using distortion effects or computed graduated fill routines: they shout 'computer art!' as loudly as any mere mechanical look. The problem is that if your art software has such features it is hard to restrain yourself from using them. I'm not advocating buying feature-free software, but keep your enthusiasm in check.

The ray tracers are intended for the production of 'realistic' scenes in full colour, not merely in 3D but showing lighting, perspective and shadowing effects. However, because everything that appears in such pictures has to be defined by the use of mathematical expressions the actual realism achievable is restricted: the best they can manage is a sort of artificial reality.

Likely to be more of use in schools or businesses than of general appeal are the CAD and presentation graphics packages. The CAD software is mainly for preparing engineering drawings, where positional precision and accuracy of shape are essential. The use of CAD in connection with DTP would be more in terms of projects and manuals rather than full-scale specifications, as the latter have special needs for printing out (such as plotters) that are not commonly used in DTP. Presentation graphics software is a way of simplifying the conversion of data to graphs or bar charts, which are widely used in businesses but which are also of great educational value: the figures can either be typed into the program itself, or they can be imported directly from a separate spreadsheet such as *PipeDream*.

The miscellaneous item in the list is more than a mere curiosity, though it might well seem somewhat specialised. *ShowPage* gives your Archimedes the power to take a PostScript file and turn it into a sprite, as long as the disk is in a format that your computer can read, of course (see Chapter 11 for a brief description of PostScript and its importance in the printing world, and also for software that will help you read other disk formats). This means that you could accept PostScript graphics from someone else's computer of a different make and incorporate them into a DTP document of your own. Given the wide range of computers that can now provide graphic output in PostScript this opens many possibilities for the interchange of illustrative material. You could also use *ShowPage* as a sort of drawing package if you are of a programming bent, by writing your own programs in the PostScript language.

You may be surprised to learn that, PostScript aside, you (or someone else) can draw pictures on other computers, with art packages native to those machines, for use on the Archimedes. This involves two intermediary pieces of software on the Archimedes: one to read the file from the disk as in the previous paragraph, and one to convert the image into a sprite. This is made possible by a public domain application called *Translator*, available from PD libraries or on Shareware disk 21 from Norwich Computer Services: as the name suggests, it translates images prepared on all manner of alien machines.

Clip Art

You may not be artistically inclined, of course, or you might just not want to be bothered with drawing your own pictures. That is all right: you can buy what is known as *clip art*, ready-made artwork, from an increasing number of suppliers for inclusion in your own work. Names to look out for (among others) are: Micro Studio, Midnight Graphics, Orion Computers, Southern Printers and SSERC. (Again, addresses are given in Appendix 2.) Clip art consists mostly of *Draw* files, mainly because they can be enlarged without loss of quality. The standard is rather variable and if not used carefully can look very repetitive and obvious, much as many small advertisements used to make excessive use of Letraset pictures about ten to fifteen years ago: they were a real give-away.

Some of the suppliers (Micro Studio and SSERC in particular) specialise in educational illustrations, such as geographical maps, science and technology symbols and historical costume; others are more general and some even cartoon-like. Capsoft offer drawn fonts and a selection of extra *Impression* frame borders.

All the DTP packages come with a selection of clip art (some rather more than others) of varying degrees of usefulness: look on this as a starting point or as something to practise with rather than as a major resource. A small selection of clip art is shown in Figure 8.4 to give you the flavour of it.

Input Devices

It is always good to have an independent source of illustrations for use in DTP: it may well be that your needs cannot be catered for by graphics software or clip art. If what you want to include as an illustration exists in a solid form (a drawing or photograph, for instance) or in electronic form (video camera or video recorder output, or a television programme) there are ways to make the conversion to a computer image.

Scanners. With a scanner you can make a copy of a picture or any design from any flat surface (even a wall!) and store it as a sprite on disk or transfer it straight to a

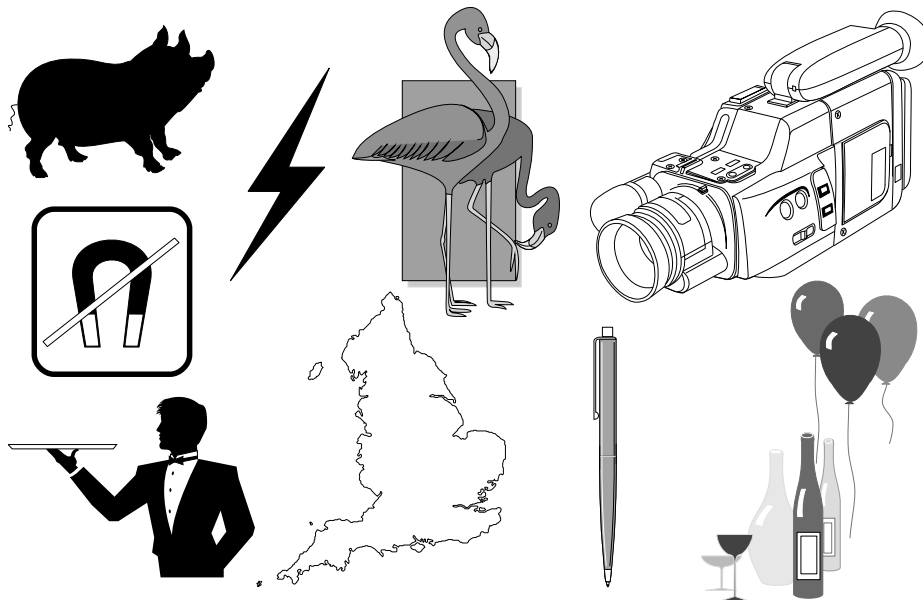
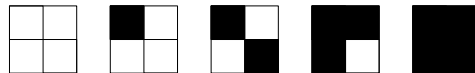


Figure 8.4 A clip art montage

document. You may find it easier to draw a picture and scan it than to draw it with *Paint* or *Draw*. Scanners for the Archimedes fall into two main varieties: hand-held and motorised. Both operate on the same principle of passing a light source over the object to be scanned and using an array of photodetectors to divide the reflected light up into tiny dots (usually at resolutions of 100, 200, 300 or 400 to the inch), which are then assembled in the computer's memory to form an image. Commonly, scanners use *dithering* to convert the grey levels in the image to black and white dots, which the software then converts back to shades of grey by *sampling* using a grid of a chosen size; the larger the number of dots from the original dot pattern you combine to form one pixel, the more grey levels you can have. For instance, a 2 × 2 grid will give you five possible levels, including white:



Larger sampling grids yield more grey levels but make the final image coarser, as more black and white dots are averaged. It is also possible to sample a grey-level sprite to increase the number of greys available, but once again at the expense of detail in the final image. For ordinary DTP work 16 or 64 grey levels should suffice. It is rare, except in professional work, for 256 greys to be required.

When it comes to printing the image, the grey levels are once again converted to dots by the printer driver (as explained at the end of Chapter 1). All this may seem rather a long-winded way of going about things, but there are good, if rather technical, reasons for it.

Unfortunately the twin processes of dithering and sampling can cause problems at the printing stage if they aren't set up in an appropriate combination: they can interact to produce a regular diagonal (moiré) pattern that may almost overpower the picture, as shown in Figure 8.5. The only way round it is to choose dither types and sampling grids that work together, and experimentation is the best way to determine this. Notice the slight loss of detail in Figure 8.5(b), which results from the lower original resolution and the larger sampling value.

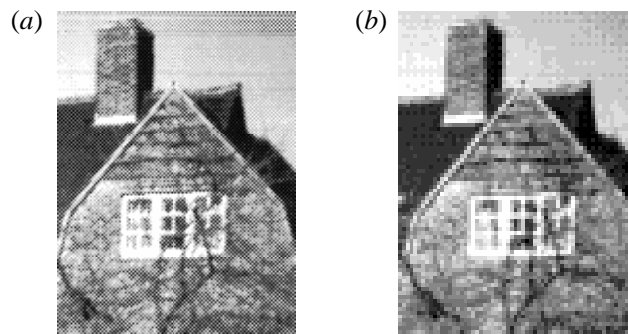


Figure 8.5 A photograph scanned with a half-width (105 mm) scanner: (a) with moiré pattern caused by conflicting scan resolution (400 d.p.i.), dither pattern and sampling grid (4×4); (b) with more appropriately chosen values (300 d.p.i., 6×6 sampling)

You may want to scan a line drawing rather than a half-tone original. In that case there is normally a switch on the scanner to prevent the dithering effect and pass an unprocessed set of black or white pixels to the computer. This can give a very accurate rendition of the original, especially if you use 400 d.p.i. resolution: Figure 8.6 shows the fineness of the detail obtainable.

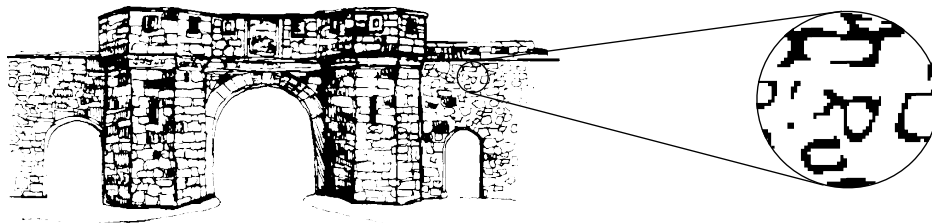


Figure 8.6 A line original scanned at 400 d.p.i.; the enlargement shows the size of individual pixels in the scan

A major problem with such scans of line originals is the large amount of memory or disk space that they take up (the picture in Figure 8.6 occupies 54K); the higher the scanning resolution the more space they occupy. Sampling can be used to cut down the sprite's size, though this reduces the amount of detail visible, as with half-tones. This isn't too surprising as the process effectively converts the black and white into grey levels, which is exactly how half-tones are stored as sprites. The amended picture in Figure 8.7 occupies only 13K after 4×4 sampling, but the averaging effect has blurred the outlines; the dots representing the grey scales are, of course, an artefact of the printing process.

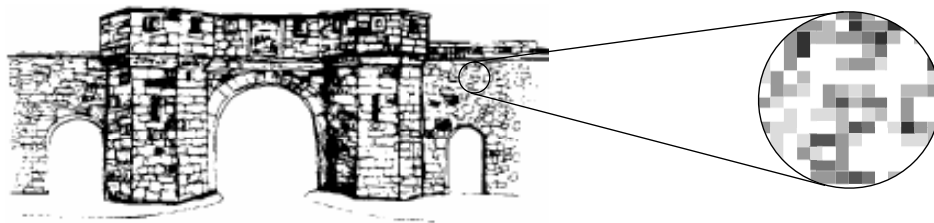


Figure 8.7 The sprite from Figure 8.6 after 4×4 sampling

Hand scanners come in two sizes: full-width, which can scan an A4 sheet, and half-width, at 105 mm wide. Naturally, the larger versions cost more: typical prices are £180–£220 for half-width and £400–£470 for the A4 size. Hand scanning requires a smooth action and a steady arm to roll the scanner over the picture evenly; most A4 scanners have an optional extra in the form of a motorised sheet feeder, at an extra cost of £120–£170. This is a good investment if you will be scanning extensively.

Half-width scanners, with software, are available from Beebug Ltd (*Scavenger*), Computer Concepts (*Scan-Light Junior*) and Watford Electronics (*Archi Mk II*); A4 scanners can be obtained from the same suppliers under the imaginative names *Scavenger A4*, *Scan-Light* and *Archi A4*. Note that A3000 versions of these, where available, tend to cost more because of the external casing required to house the expansion card.

In the above I have been considering only monochrome (black-and-white) scanners, mainly because colour scanners are much more expensive (upwards of £600; suppliers are Irlam Instruments (*iMage*) and Clares (*ArcScan*)) and are therefore really only within the reach of professionals. You can, of course, scan colour photographs with a monochrome scanner: with suitable adjustments of the brightness and contrast controls (one on the scanner itself and the other usually in the software) quite acceptable results can be obtained.

The day of professional colour on the Archimedes is approaching now that the 256-

colour barrier has been broken: The Serial Port have developed a graphics enhancer (the PCATS Graphics Enhancer expansion card; it costs about £200) that vastly increases the size of the palette from which you can choose your 256 colours in the Desktop: the range is 16,777,216 colours instead of 4,096. Even if you don't need such a choice of colours you could have a set of 256 different greys, which would make your half-tones magnificent.

Newer monochrome scanners don't use the dithering technique to generate the half-tone effect; instead they output grey-scale sprites direct. The 200 d.p.i. resolution gives far better results than a dithered 400 d.p.i. scan. On the down side, the file sizes tend to be colossal. Irlam and Computer Concepts supply scanners of this type.

Digitisers. What scanners do for the printed page, digitisers do for the video image. In other words, digitisers take a picture from a video source and convert it into a sprite. This is invaluable for providing a permanent record of ephemeral events and can be of benefit in almost any situation (school, home or business); the possibilities are practically limitless. Digitised images have an immediacy, a vitality that can be missing from scanned photographs, though in principle you can't get as fine a resolution as with a scanned image because a digitised one is limited by the detail obtainable from a video source such as a television picture. This may restrict the final size at which you can print such images. Figure 8.8 shows a couple of digitised pictures by way of example.

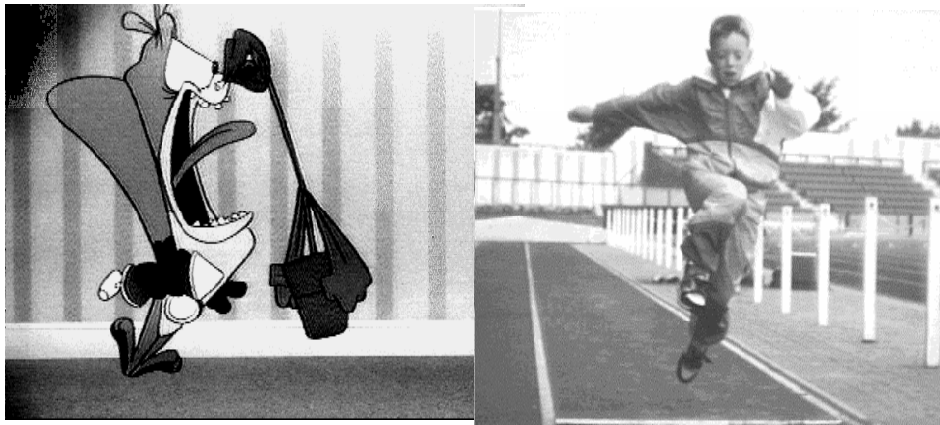


Figure 8.8 A pair of digitised images (courtesy of Mike Cook and Wild Vision)

As with scanners, digitisers come in both monochrome and colour versions. Monochrome digitisers cost in the region of £150–£170 (try Brainsoft's *Multipod Professional* from Technomatic, or the *Archimedes TV Digitiser* from Watford Electronics), and colour ones range between about £250 and £400 (*The Pineapple Colour Video Digitiser* comes from Pineapple Software, *Snapshot* from Lingenuity,

Hawk V9 from Wild Vision and *techno-I* from Technomatic). *Lingenuity* also provide a colour converter for the Watford digitiser.

One thing to look out for is 'real-time' frame grabbing, where the software and hardware work fast enough to feed the computer an image as fast as it is built up on the video device. This means that you can freeze a single frame of a moving sequence. Without this facility you are limited to using either still or slow-moving pictures, or input from a video recorder in freeze-frame mode (and by no means all recorders can freeze a frame without judder or bands of noise across the picture).

Once the picture has been grabbed it can be manipulated in many ways: the software provided with some digitisers offers features like enhancement, smoothing, reflection and defocusing. Full-colour screens can be memory-hungry; for DTP work you will probably want to convert them to monochrome, or at least to a mode with fewer colours. For this you can use *ChangeFSI* from Wild Vision, which is a clever piece of image manipulation software for converting sprites from one mode to another, or from colour to monochrome, with as little loss of quality as possible. *ChangeFSI* can also, incidentally, translate images from other computers, just as *Translator* can, but *Translator* doesn't have the powerful mode-conversion features.

Storage Troubles. As already mentioned, scanned and digitised images are greedy for space, and if you get carried away with them you may quickly run out of storage room on your disks, or you may fill so many that you have a hard time keeping track of them. There are two ways to reduce this problem (apart from exercising strict discipline in either not producing so many images in the first place, or culling those you don't need at regular intervals): data compression and data conversion.

Data compression programs take the original files and simply squeeze them into a smaller space; the information itself is retained but occupies less room because it is in an encoded form, and must be decoded (expanded) before it can be loaded back into your Archimedes. A program for doing this is *Spark*: it can squash suitable files to half normal size or less as well as expanding them again. *Spark* is available from David Pilling (its author). Another compression program, specifically for use with sprites, is published by Alpine Software.

By data conversion I mean the turning of sprite files into *Draw* files: this is a neat trick, and can save a fair amount of space because *Draw* files tend to be smaller than sprites for simple subjects. As it involves the manipulation of graphical information, though, we shall look at it in the next section.

Graphics Manipulation

Software for the Job

So, you have the pictures you want but they aren't quite right in some way? Some alterations can be made after you have placed them within your DTP document, but some changes may be needed before then. Therefore, as a prelude to looking at the how and what of DTP graphics themselves, let us see how they can be adjusted beforehand.

The first and most obvious pieces of software for altering graphics files are *Paint* and *Draw*; for the most part I have stuck with them in preparing the illustrations for this book, and *Paint*'s 'Get screen area' facility was used extensively. *Paint* is good, if a little slow and restrictive, for making fairly basic modifications to sprites: rotation can be tiresome as it may run out of room, often takes a long time and can lead to cumulative errors. However, it is quick and straightforward for changing a sprite's size by adding or deleting rows or columns, or for grafting different images together.

Draw is a much underrated piece of software, probably because one tends not to value what comes free. I don't intend to go into the details of how I prepared Figures 8.6 and 8.7 from scanned sprites using only *Paint* and *Draw*, partly because it would be tedious to describe and partly to maintain a small air of mystery, but remember that *Draw* files can contain sprites as well as *Draw* objects proper. However, *Draw* can't rotate sprites, just as it won't rotate text.

This brings us to the promised look at a way of turning sprite files into *Draw* files. The piece of software in question is *Tracer* from Midnight Graphics, and the process offers two advantages: by tracing round a sprite and converting it to *Draw* format the image can then be enlarged indefinitely without it degrading into the jaggedness of individual sprite pixels, and secondly the resulting *Draw* file often needs less storage space. The success or otherwise of the tracing depends on how well you have set up the program's various options and also on the shapes to be traced themselves: straight lines meeting at right angles, for example, are not too easy to reproduce. By

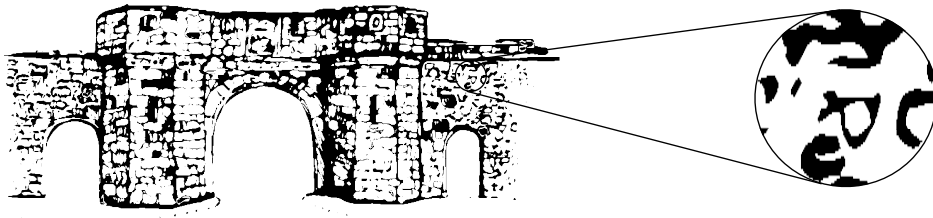


Figure 8.9 The picture sprite from Figure 8.6 traced by *Tracer*; the jagged edges in the inset arise from its having been grabbed from screen as a sprite

a curious chance, the picture in Figure 8.6 was so complex that its corresponding *Draw* file, shown in Figure 8.9, occupied more than twice as much space (126K) as the original sprite (54K). This was an unusually hard test: the conversion took several minutes, which is scarcely surprising, and it was amusing to see my Archimedes having to struggle to redraw such an enormous object on screen in *Draw* (and even in *Impression*). It was so-o-o *slo-o-o-w*. Happily, you are unlikely to want to go to such extremes, and the potential space savings are considerable. Also, even large *Draw* files print more quickly than sprites.

With some care you can also use *Tracer* to render coloured sprites into *Draw* images, but a great deal more human intervention and manual operation are required.

The graphics manipulation packages *Poster* and *DrawBender*, mentioned in the previous chapter, are just as capable of distorting pictures as text. However, because neither of them can act on sprites, their greatest potential can only be realised by using one or the other in concert with *Tracer*.

Graphics in DTP

Importing Graphics. By comparison with the section on importing text, the following description of how to import graphics into DTP will be so short that if you blink you will miss it. The principle is the same as for text: just drag a file or a Save icon over a selected frame, which must either be new or already have graphics in it. You can't import graphics into a text frame, even if it is empty (*Ovation* has a separate frame type for graphics anyway, and in the other DTP packages importing text into a new, or null, frame turns it into a text frame).

The different packages treat the newly imported picture slightly differently: *Acorn DTP* stretches it to fill the rectangle of the frame regardless of the shape (the *aspect ratio*, or the proportion of height to width) of the original, whereas *Impression* and *Ovation* make it as large as possible within the frame while maintaining the original aspect ratio. In this respect *Acorn DTP* is at a disadvantage because you may not want to stretch the picture: we shall see shortly why this is undesirable.

Another point of difference is in the treatment of text imported into *Draw* (see Chapter 4). Of the three DTP packages only *Ovation* displays it at all; the others simply show a blank space where the text area was. This may be important if you are assembling DTP documents from items sent to you by other people, so you should keep it in mind and examine all *Draw* files that you receive, just in case. If it happens to you, you can use *Draw* to export the text area as a text file, load it into a word processor to edit out the formatting commands and then re-import it into your DTP program as a text frame with a transparent background, over the top of the imported graphic.

Cropping and Resizing. Once the picture is in a frame you can move it about on the page and generally alter its appearance with the techniques of *cropping* and *resizing*. The various methods of resizing a graphic and/or its frame are explained in the software manuals so there is little point in duplicating that information. It can be done as simply as dragging a frame corner with Adjust, or you may have to call up a dialogue box and make adjustments. The outcome is a frame of the size you want, filled either widthways or from top to bottom with the original picture, assuming that you haven't altered its aspect ratio.

Cropping a picture allows you to choose a part of it and ignore the rest. Basically it involves making the frame smaller than the graphic (or the graphic larger within the frame) to cut off the unwanted portions: any part that lies outside the frame will not be shown, although the whole picture is still held in memory. You can therefore change your mind about which part you want to be visible without having to re-import the whole thing. In some programs you perform cropping via dialogue boxes, in others there are keyboard shortcuts as well.

Stretching. As already mentioned it is possible to change the aspect ratio of a graphic from its original value; indeed, with *Acorn DTP* it is hard to avoid. This has the effect of stretching the picture horizontally or vertically, which can be useful in very rare circumstances to fill a difficult hole on the page.

However, for an illustration that shows anything from the real world itself or derived from it (that is, anything representational as opposed to abstract or artificial) it is always better to crop an area to a desired shape rather than stretch the whole picture. The reason is that the eye is quite sensitive to even small distortions of reality, and unless the effect is part of your message it will serve only to distract or confuse the reader. Cropping also lets you emphasise a part of the whole illustration by 'zooming in' on essential detail and omitting extraneous areas.

Take a look at Figure 8.10: (a) is the original picture (for historians, it shows the Mexican rebel Melquiades Melendez and his guerrillas at Culiacan in April 1912, so the image is a scanned photograph rather than a digitised contemporary video) and (b) is how *Impression* or *Ovation* would initially fit it into the frame shown. In (c) the whole graphic has been made to fill the frame by stretching, in (d) the width only has been cropped to show as much as possible of the original, and in (e) the cropping has focused on a smaller portion of the picture. See how different each one looks and the range of messages that may be conveyed (in (d) he seems to have fewer followers than in (a), but what the message is in (c) I am not sure).

Panning and Rotating. *Impression* and *Ovation* have facilities for panning and/or rotating a graphic within its frame: these are useful for applying final touches to a picture when it is actually in the document, as you may not know until you see it in its context exactly how it should look. They can be real time-savers in the graphic

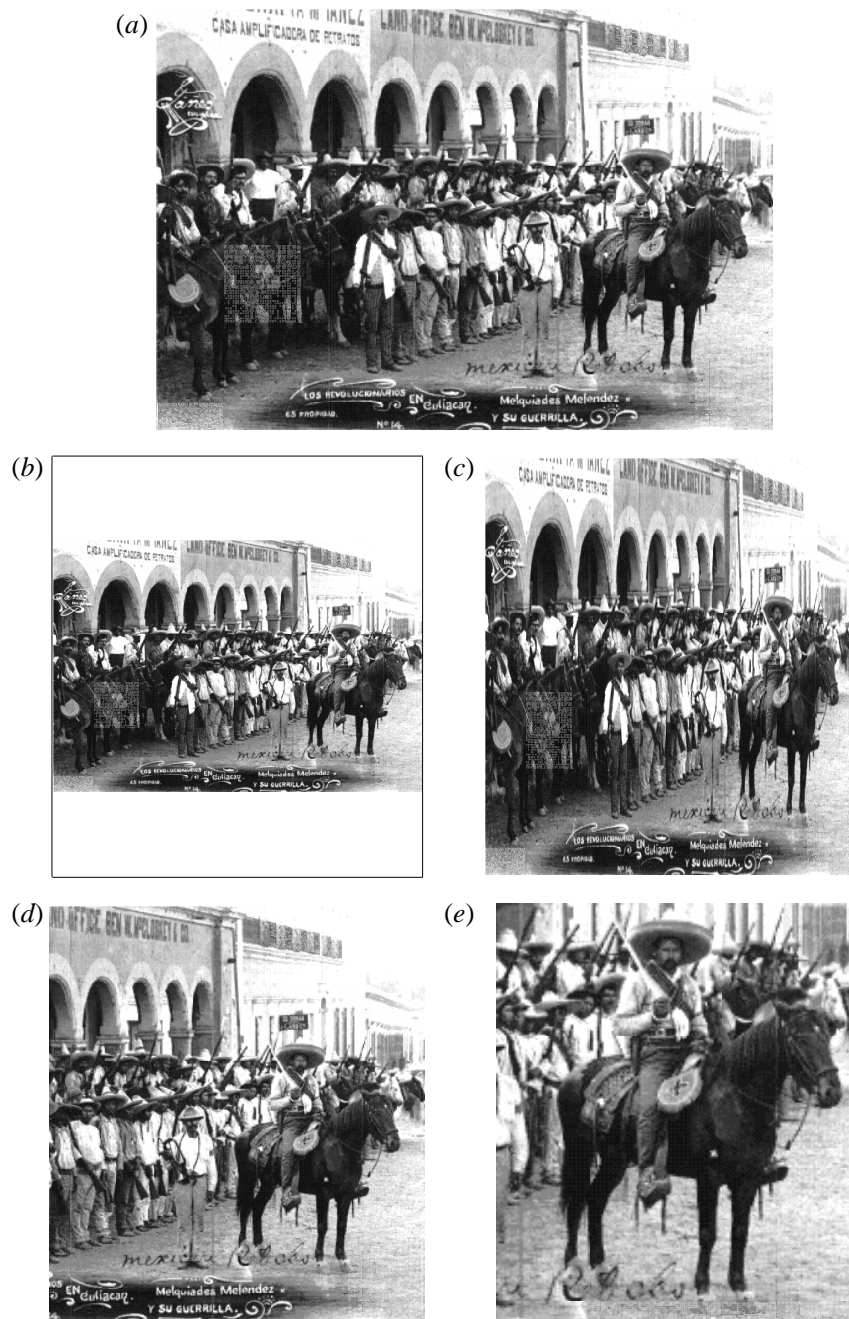


Figure 8.10 Different ways of looking at the same picture

phase of your work.

Panning refers to the ability to 'grab' the picture and move it around inside the frame, which is a lot quicker than using the frame edges for cropping. *Ovation* and *Impression* have this facility. Rotation, which is available in *Impression*, is especially useful for sprites because of the limitations of *Paint* (it is slow and runs out of memory). The part of each of these programs that does the redrawing is much faster than *Draw*, too, and (unlike in *Paint*) the rotation is done in such a way that the original is unaltered in memory, so tiny errors don't accumulate as different rotations are applied successively to the same graphic. To be able to rotate sprites, *Impression* has to have a special module loaded; this is selected as 'Enhanced graphics' from the Preferences dialogue box.

There is one thing to watch out for when rotating *Draw* images in *Impression*. We have already noted that text areas disappear; note also that text typed into a *Draw* file isn't rotated, as Figure 8.11 shows. It stays horizontal, which can sometimes be an advantage; however, if you want to label anything and have the labels as an integral part of the picture, use *FontFX* or any other text-to-*Draw* converter.

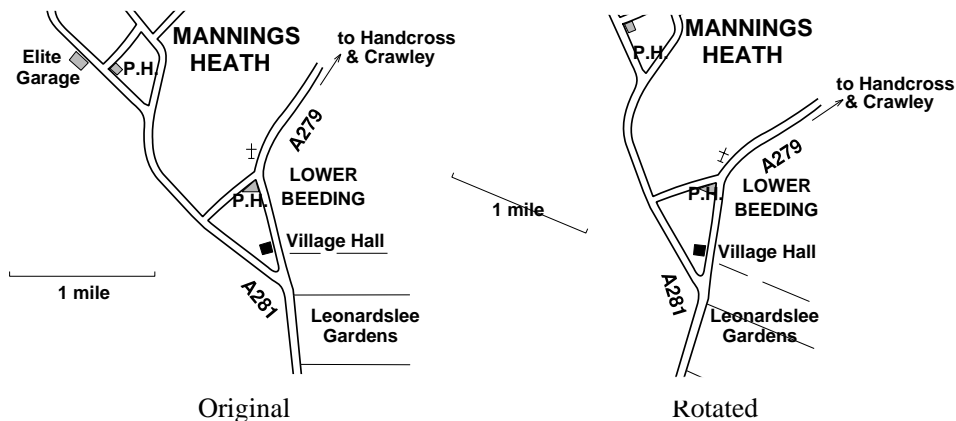


Figure 8.11 Rotating a *Draw* file containing text: the picture rotates, the text doesn't

A Memory-saving Tip. Here's a quick *Impression* hint for saving memory when using the same graphic more than once in a document. The sprite of the Mexican bandits (and the *Draw* file in Figure 8.11) was imported only once and then copied into the other frames by the same technique as used to flow text: selecting the original frame and clicking Adjust on the others. In this way the graphic needs storing only once in memory; all the other versions aren't so much copies as separate facets of the original, treated independently.