Activity 2

Color by numbers—*Image representation*

Age group	Early elementary and up.
Abilities assumed	Elementary counting, and some concentration. Activity 1 (Count the Dots) is helpful, but not essential, preparation. Experience with graphing is also useful.
Time	20 minutes or more.
Size of group	From individuals to the whole class.

Focus

Representation.

Coloring.

Pictures.

Summary

Computers are often used to store drawings, photographs and other pictures. This activity shows how pictures can be represented efficiently as numbers in a computer.

Technical terms

Raster images; pixels; image compression; run-length coding; facsimile machines.

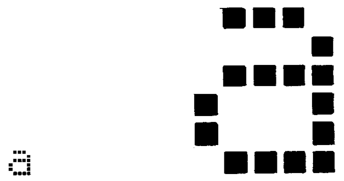


Figure 2.1: The letter “a” from a computer screen, and a magnified view showing the pixels that make up the image.

Materials

Each child will need:

a copy of the blackline master on page 25, and

a pencil and eraser.

You will need:

an overhead projector transparency of Figure 2.1 (or draw it on the classroom board).

What to do

1. Discuss what facsimile (fax) machines do (arranging for the children to send and/or receive faxes would be excellent preparation for this activity). Ask for other places where computers store pictures (for example, a drawing program, a game with graphics in it, or a multi-media system).

   Explain that computers can really only store numbers (if they have done the activity on binary numbers then they will already have some appreciation of this). Have the children suggest how a picture might be represented using only numbers.

2. Demonstrate how images are displayed on a computer screen, as follows:

   Computer screens are divided up into a grid of small dots called *pixels*. The word pixel is a contraction of “picture element.” On a black and white screen, each pixel is either black or white. Figure 2.1 shows a picture of a letter “a” that has been magnified so that the dots are visible. (This image should be shown on an overhead projector, or drawn on the board.) When a computer stores a picture, all that it needs to store is which dots are black and which are white.

3. Using the image of Figure 2.1, demonstrate how pictures can be represented by numbers on the blackboard, as follows:

   Begin by writing down the number of consecutive white pixels in the first line of the image (in Figure 2.1 there is only one white pixel at the start of the first line). Then write

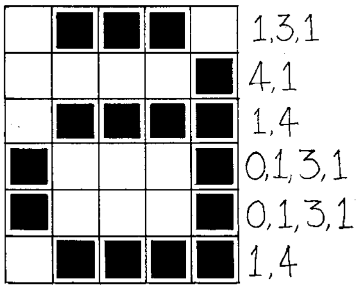


Figure 2.2: The image of Figure 2.1 coded using numbers

down the number of consecutive black pixels (three), and so on until the whole line has been coded. Thus the first line of Figure 2.1 will be represented as 1, 3, 1. Each row of pixels in the image is transcribed using this method. For example, the second line of Figure 2.1 is represented by 4, 1, since it has four white pixels followed by one black one. Figure 2.2 shows the final representation of Figure 2.1. Notice that lines beginning with a black pixel have a zero at the beginning to indicate that there are no white pixels at the start of the line.

There may be some confusion with this demonstration if you use a chalkboard, because coloring in a pixel makes it white rather than black!

4. Have the children "decode" the pictures on the worksheet on page 25.

   Hand out the worksheets and get the children to decipher the images there (see Figure 2.3 for the completed images; Figure 2.4 contains a reduced version which makes the pictures clearer). The "test" picture on the right is the easiest, and the "being" on the left is the most complex.[1] The squares on the grid should be colored in completely, and made as dark as possible. If the squares on the blackline master are too small, you can use quad-ruled paper. It is important to use pencil, as mistakes are easily made! Children should mark each line of numbers as they finish with it. Some may prefer to circle all the numbers that represent black pixels to help them keep track of where they are up to—it is very easy to get confused about which numbers are for black and which are for white.

   Note that each line always begins with the number of white pixels. If the first pixel is black, the line will begin with a zero.

Variations and extensions

The image will be clearer if the children draw on a sheet of tracing paper on top of the grid, so that the final image can be viewed without the grid.

Instead of coloring in the grid the class could use squares of sticky paper on a larger grid, or place any objects on a grid. The codes could even be used for a cross-stitch pattern.

[1] The images are taken from the excellent children's drawing program *KidPix*, by Brøderbund Software.

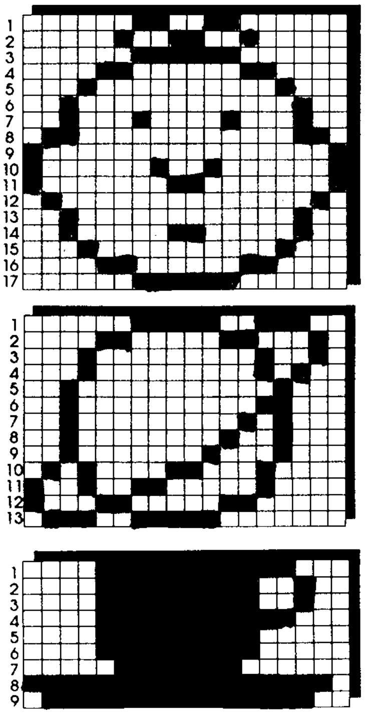


Figure 2.3: The completed images

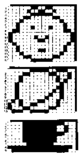


Figure 2.4: The completed images reduced

Children can try drawing their own pictures on a grid (or try to copy one from a computer screen). They can code the picture as numbers and give it to friends to decipher.

In practice there is usually a limit to the maximum length of a run of pixels because the length is being represented as a binary number. Ask the children how they would represent a run of twelve black pixels if they could only use numbers up to seven. (A good way is to code a run of seven black pixels, followed by a run of zero white, then a run of five black.)

The method can be extended to colored images by using a number to represent the color (e.g. 0 is black, 1 is red, 2 is green etc.) Two numbers are now used to represent a run of pixels: the first gives the length of the run as before, and the second specifies the color.

What's it all about?

The simplest way to represent a black and white picture on a computer is to represent white pixels with a zero (say) and black with a one. However, it is common for images to contain large blocks of white pixels (especially in the margins) and runs of black pixels (e.g. a horizontal line). Facsimile documents commonly have about 100 pixels per inch, so a row of white pixels at the top of a 7 inch wide page could take 700 bits of storage.

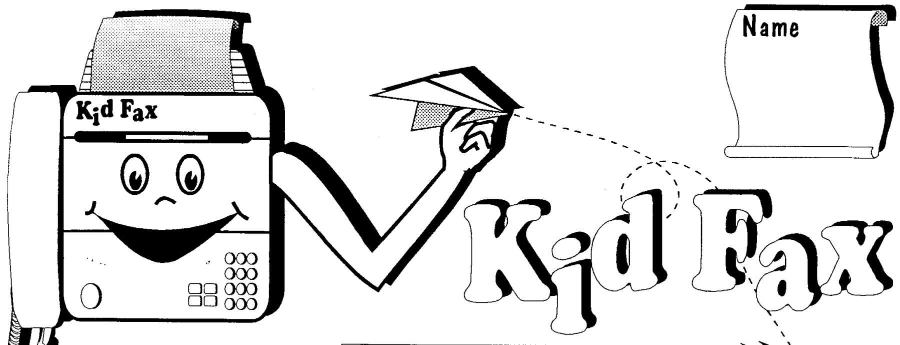
The "run-length coding" that we used in the activity above is much more efficient. It represents the run-length of 700 using the binary system (see Activity 1), which can be done in 10 bits. This example is a little extreme, but nevertheless significant savings can be made.

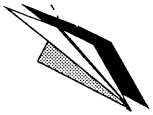
Saving space using techniques like this is called *compression*, and it is crucial to the viability of working with images on computers. For example, fax images are generally compressed to about a seventh of their original size. Without compression they would take seven times as long to transmit, which would make fax a much less attractive means of communication. Images stored on computer disks are often compressed to a tenth or even a hundredth of their original size. Without compression there might be room for just a handful of images on the disk, but with compression many more images can be stored. The advantage is even greater when storing moving pictures, which normally involve 25 or more images each second. The compression method used in this activity is related to the method used on most common fax machines. A host of other methods are also available. The ones that give the best compression are "lossy"—they change the image very slightly so that it can be stored much more efficiently.

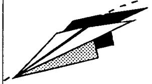
Is there a down-side?—can "compressing" an image ever *expand* it? Well, yes. A checkerboard image in which pixels alternate black and white would be much more efficient to store using one bit for each pixel than using one number for each pixel, because it will be necessary to represent the number as several bits to allow for the possibility of longer runs. Although this particular case is unlikely, some real images—such as halftone images, like illustrations in newspapers, that are made up of lots of tiny dots—do not compress well and may even expand. It is sometimes necessary to tailor the representation method to the kind of image being represented.

Further reading

Representing images on computers is discussed in depth by Netravali and Haskell in *Digital pictures: representation and compression*. The standard method for coding on fax machines is described by Hunter and Robinson in a paper published in 1978 entitled “International digital facsimile coding standards.” A more recent standard for coding images is described in the book *JPEG: Still Image Data Compression Standard* by Pennebaker and Mitchell.



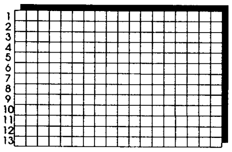


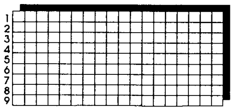


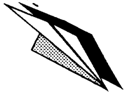
	Being
Row 1	6, 2, 2, 2
2	5, 1, 2, 2, 2, 1
3	6, 6,
4	4, 2, 6, 2
5	3, 1, 10, 1
6	2, 1, 12, 1
7	2, 1, 3, 1, 4, 1, 3, 1
8	1, 2, 12, 2
9	0, 1, 16, 1
10	0, 1, 6, 1, 2, 1, 6, 1
11	0, 1, 7, 2, 7, 1
12	1, 1, 14, 1
13	2, 1, 12, 1
14	2, 1, 5, 2, 5, 1
15	3, 1, 10, 1
16	4, 2, 6, 2
17	6, 6

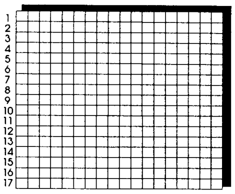
	Planet
Row 1	6, 5, 2, 3
2	4, 2, 5, 2, 3, 1
3	3, 1, 9, 1, 2, 1
4	3, 1, 9, 1, 1, 1
5	2, 1, 11, 1
6	2, 1, 10, 2
7	2, 1, 9, 1, 1, 1
8	2, 1, 8, 1, 2, 1
9	2, 1, 7, 1, 3, 1
10	1, 1, 1, 1, 4, 2, 3, 1
11	0, 1, 2, 1, 2, 2, 5, 1
12	0, 1, 3, 2, 5, 2
13	1, 3, 2, 5

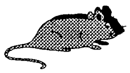
	Test picture
Row 1	4, 11
2	4, 9, 2, 1
3	4, 9, 2, 1
4	4, 11
5	4, 9
6	4, 9
7	5, 7
8	0, 17
9	1, 15











Instructions: *Use the numbers to color in the squares. There is a row of numbers for each line in the picture. For example, the line 4, 9, 2, 1 means to leave 4 squares empty, color in 9, leave 2 empty, and color in the next one.*