Analog and Digital Data

by PC Tech | Sep 24, 2011 |

Let’s not forget that computer data is digital: that is: it is a series of zeros and ones, and nothing else. It’s a point easily forgotten. Computers, we know, have an extraordinary power to manipulate numbers at this digital level, which means (with a little number base work) they are perfect for dealing with any numeric data we throw at them. We now use them routinely to perform phenomenal calculations, but there is a basic problem with a great deal of the data we ask them to handle, and it stems from a simple fact: the world we live in is analog, not digital.

This was an issue long before computers existed, as people tried to find ways to discuss their analog world with each other. The trouble is, our analog world is so subjective that it poses some curious difficulties when it comes to communication. Though it may have certain rhythms or patterns, analog data is continuous, and has no intrinsic calibrations for us to use to communicate with. For example, we may know that the temperature varies between hot and cold with the time of day and seasons. But hot and cold are not precisely defined, are open to subjective judgement, and so can’t be communicated with any real accuracy: today is hot, and so was yesterday – but which was the hottest? When is it warm? Cool? Cold? Is warm to someone in the Scottish hills the same as warm to another person on the Florida coast?

The solution humans adopt for these analog issues is to introduce a means of measuring. The means of measuring must have a consistent means of sampling, but otherwise is entirely arbitrary. However, this measuring has no value unless it is shared and widely adopted. Then, it becomes an invaluable tool.

For instance, continuing with our temperature example, the Celsius scale of temperature measurement defines the freezing point of water as 0 and the boiling point of water as 100. It then divides the intervening points into 100 equal units, and extrapolates these units for temperatures below 0 and above 100. An eminently sensible system, those using it accept it as completely natural. However, beyond being based on water’s freezing and boiling points there is nothing natural about it. Importantly, though, it can be consistently sampled, has been widely adopted, and therefore provides a basis for communication and science.

Interestingly, the measuring is often done in an analog to analog way, but with a calibration imposed. For instance, when measuring temperature, we watch the expansion of mercury up a glass tube. The mercury expansion is an analog event, but the Celsius measurements are not. A sundial or even a clock hand represent time in an analog way, but we impose our measures of time on the rotation. Speed, sound, mass, gravity, acceleration, voltage, current and so on; all are measured by testing against a second analog signal which is affected by the first and applying a calibration.

With the digital computer, the calibration is simple and predetermined: binary. By whatever means, everything a computer deals with must ultimately be reduced to zeros and ones. So now, on to the pictures.

Well, the fact is that image data is not only analog in nature, it has many levels of complexity. With temperature, data can be measured in a linear fashion as it has only one dimension of variance: cold to hot. Applying numerical values to temperature, therefore, seems logical: low to high. An image, on the other hand, has little that easily lends itself to numeric representation. Nonetheless, the way that images are stored digitally is still based on numerically translated sampling and measuring techniques, just taken on to another level.



Let’s take a square image, like this one for instance.

We’ll overlay it with a grid that is 10×10. Notice there is no unit of measurement used at this point, because it isn’t necessary – yet. Each square in the grid we can say is a picture element, but they’re referred to as pixel for short.



Each pixel contains a small area of our image, and forms the basis of digital image representation. Starting at the top-left pixel, we calculate an average of the color within it.

We continue through each pixel left to right, top to bottom, until every pixel’s color has been sampled.

Clearly, if only for simplicity’s sake, we’ve used a very low resolution for sampling our image. On the next page we’ll discuss some of the implications of sample resolutions for digital images. Seen side by side, it’s clear that the original image has been, to put it mildly, somewhat obfuscated by the sampling process.



Although in our example the difference is very noticeable, the loss of definition is an inevitable part of the image sampling process, no matter how fine the resolution. However, the fact is that the human eye is only capable of so much distinction, both of detail and of color. In fact, we fool our eyes all the time with any screen image by simply changing the screen more quickly than our eyes can detect, even when the image is apparently still. This gives some leeway when it comes to choosing a sample resolution for an image.

We’ll come back to that later in the article, but for now we’ll take a look at these two images at a much reduced size.



Digital image capturing devices such as scanners, digital cameras and digital camcorders are designed to be used with this knowledge in mind. Essentially, when sampling a very large image that is going to be shown in a smaller form, the sampling resolution need not be so high. If, on the other hand, a relatively small image is going to be blown up to a larger size, then the sampling resolution needs to be as high as possible. Otherwise, when the image is expanded, the pixilation seen in the above images will occur, and there’s very little that can be done about it. However, don’t forget that there is only so much detail that a human can use, and oversampling leads to bloated files full of data that is more or less invisible.

As an image is captured and a grid of color samples built up, each grid location’s color needs to be given a representative number. Like a painter, when dealing with images the computer uses a palette. The palette contains the color information for the picture, and, like the painter’s palette, forms the basis for creating the image. The image data can be developed from one of many existing predefined computer color palettes, or, as in this example, a palette of colors can be constructed from the colors found in the image, ascribing each color a numerical value as we go.



Matching Color Palettes?

There are many standard palettes in computing, for instance the 216 web safe colors, the Windows palette, and the 16 bit RGB true colors. If necessary, a palette can be uniquely defined for a particular image, but the same palette must be used when displaying the image as used when creating the image. If the palettes match, the result will be the same on any computer – at least in theory.

The basic task is to go through each square and turn our image into a grid of numbers. Once that is done, if we take each number in sequence, left to right, top to bottom, and write the numbers sequentially we create a stream of byte by byte data.

Computer data is binary, so we translate byte by byte into binary code. When that’s done, our stream of data looks something like this:

101000010111110101000111100011000101111000101101011010110011100100010101011000101000010000000010

Once we’ve completed this bit by bit data stream for the whole image it can be stored in the computer’s memory. The computer now has data it can handle, and we have succeeded in creating a digital version of our picture. When it comes to displaying the image, as long as we know the system by which the data was created, which number fits which color and where for a particular set of image data, the digital information can be translated back to color to display the image as desired on screen.