Picture this
On the proper use of graphics in Web publishing

Martin Vermeer

May 3, 2002

Publishing on the Web is in many ways quite different from paper publishing, and as a young art, few people that practice it have received any formal training in doing so. Especially the inclusion of graphics appears difficult. Having been involved myself in Web publishing I have had opportunity to become aware of the most popular pitfalls.

It is unfortunately easy to prepare pictures that

- ◇ look awful
- ◇ fall way short of the intended resolution or image sharpness, yet
- ◇ consume inordinate amounts or disk space.

First off: contrary to textual documents, pictures are *intrinsically big*. A picture is worth a thousand words, but also, unfortunately, easily occupies a lot more than a thousand words' worth of disk space — or would, if we didn't use one of several tricks available for achieving much greater economy in graphics storage. This is important as Web pages, contrary to paper print, are to be *downloaded* by the user into his browser, which may become an unpleasant wait if pains are not taken to keep graphics small and the user has a slow connection to the Internet.

The simplest, most straightforward way to represent a picture in computer readable form is as a *grid* of *pixels*, picture elements: *raster graphics*. This is the way a computer display screen works. On a typical monitor screen, you may have 75 DPI (dots per inch); if the size of your display is 1280 pixels (dots) wide and 1024 pixels high, this means that your screen size in inches is $\frac{1280}{75}$by $\frac{1024}{75}$, or 17×14 inches approximately[1]. The storage requirement is obtained as 1280×1024 pixels ≈ 1.3M pixels.

[1]You get the screen diagonal size (which manufacturers announce) by Pythagoras: $\sqrt{17^2 + 14^2} = 22$ inches.

Now, the pixels themselves may have different storage sizes. If your picture contains only black and white and not even grey (or black and green, or more generally only "on" and "off", two different states) we speak of a *monochrome* display. In this case you need ony one *bit* for every pixel, containing a 1 for white or a 0 for black. If you want *greytone*, however — say, 256 different levels of grey — you will need 8 bits, or a *byte*, to represent the value of a single pixel. This adds up to 1.3 MB (megabytes) of storage requirement.

And if you want colour, multiply this by (typically) three[2].

We have been talking this far about a screenful of graphics. Print it on paper, at a modest resolution of 300 DPI (quality laser printers do better) and you get a picture of size 4.3×3.4 inch or 10.7×8.5 cm. This picture will look sharp to the human eye. It will not even fill half a page, whereas half a page — or a computer screenful — of text would only require something like 1 kB of storage, one-tenth of one percent of the (greytone) picture considered here!

Conclusion: pictures are *big*.

Pictures do not have to be very big on disk, however.

Already in the early days of the Web, when many people had slow dial-up lines, it was found advantageous to *compress* images. Popular compression formats are GIF, JPEG, TIFF and the newer PNG. Raster image compression techniques fall into two categories: *lossless* and *lossy*.

- ◇ The lossy techniques, such as JPEG, slightly modify or "nudge" the individual pixel values in order to obtain a better compression ratio, but losing something in the process. These techniques are

[2]Often used is the RGB (red, green, blue) colour representation, where the brightness value of each of those primary colours is given by a single byte.

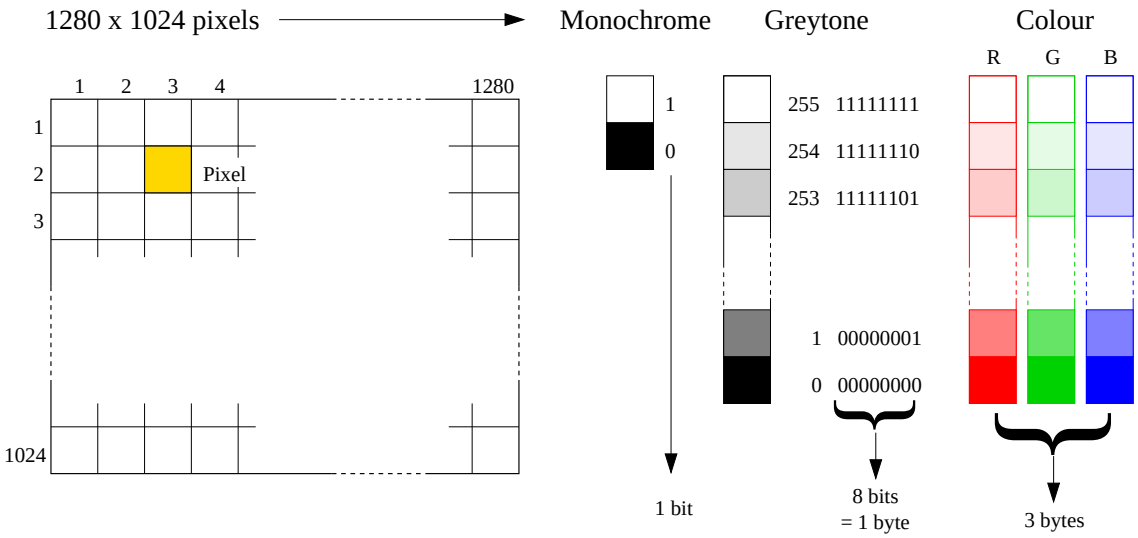

Figure 1: Raster graphics

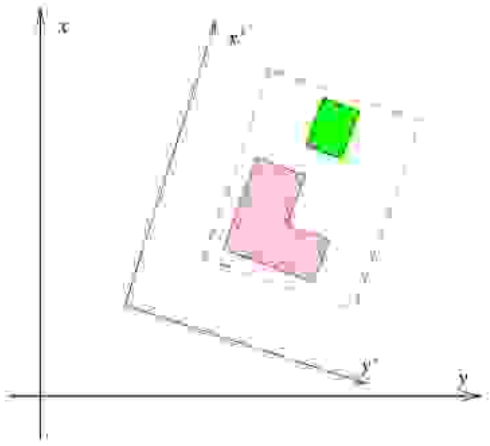

Figure 2: Artefacts in a lossily compressed picture (exaggerated)

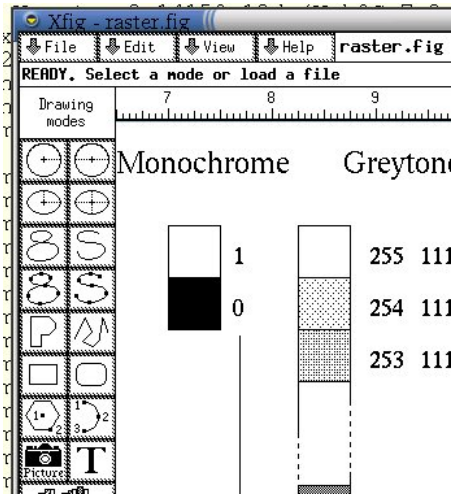

Figure 3: A popular drawing program (image file size JPEG 42 kB, GIF 41 kB)

worth using with *photographic imagery* and the like, which are always slightly imperfect analogue representations of reality anyway.

- ◇ When your pictures are more like *diagrams* or technical drawings, lossy compression techniques tend to produce ugly-looking and disturbing *artefacts*. For these, rather use lossless compression techniques such as GIF or PNG.

Representing a technical drawing as a grid of pixels is actually a *bad idea*. Many of the popular drawing programs allow you to edit a drawing *logically*, by moving straight lines, circles, polygons and other picture elements around, scaling or reshaping them, putting in text labels etc. etc., but all the time keeping these logical elements *separate* from each other, also in the on-disk storage file. Compare this to a raster graphics file, where everything is just pixels even if the human eye discerns circles and straight lines and legible text.

The conversion of such a "logical" picture, or *vector image*, to a raster image is a one way process, a trap door if you like, the wholesale destruction of valuable information. The way back — *vectorization* — poses a

challenging artificial intelligence problem.

Popular vector graphics formats are: EPS (encapsulated PostScript), PDF (portable document format) and the still very young SVG.

EPS and PDF are very similar, originally produced by Adobe but exhaustively documented and widely used in industry; unfortunately there is no straightforward way to include them in a Web page without requiring the reader to download a special "plug-in" for his browser.

Properties of vector image formats are:

- ◇ parsimonious in storage space, even more so than the brute-force compression used on raster graphics;
- ◇ freely *scalable*. You can blow up the picture to absurd sizes without seeing "fat pixels" or sharpness degradation.
- ◇ For this scaling to work, the text fonts included in vector graphics should themselves be vector graphics based, so-called *outline fonts*. These can be PostScript type 2 or TrueType fonts. Raster fonts will look ugly and not scale well.
- ◇ Vector graphics preserve the *logic* of the picture, the intentions of the draftsman, in a way raster graphics cannot. This alone makes it the solution of choice if choice exists.

Rules of thumb for Web graphics

1. If you need the stepless, unrestrained scalability that vector graphics give you, use PDF. The price you pay is requiring your users to install the AcroRead plug-in; as a bonus you get printed-quality documents on-screen.
2. If requiring a separate plug-in is not acceptable, you are (today) pretty much stuck with raster graphics for your Web pages.
   (a) Use JPEG for photograph-style imagery, and GIF (or the more and more popular alternative PNG) for diagram-style graphics.
   (b) For large pictures, provide a small "thumbnail" to click on to get the full picture. Users behind a slow line will be grateful.

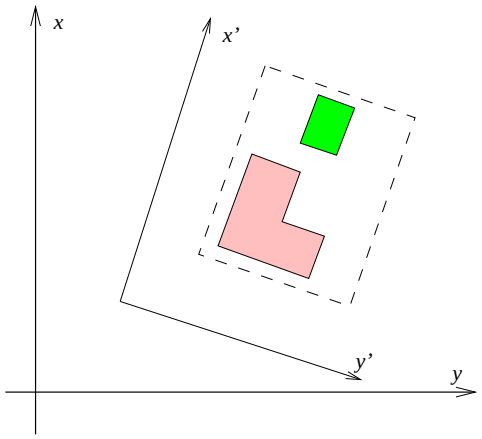

Figure 4: First example.

EPS:	4 kB	PDF:	1.2 kB
JPEG:	8 kB	GIF:	4 kB

   (c) No picture file, whether vector or raster, should *ever* occupy more than 10% of what the picture size in pixels would imply. A 512×256 pixel colour picture that's over 38 kB in size[3]? You're probably doing something very wrong. File sizes of 1 – 5% are more typical. See the example pictures for illustration; you can visually interpolate reasonable sizes for your own pictures depending on size and degree of detail.

3. *Don't ever use EPS or PDF raster files!* These are possible, even easy, to create accidentally, e.g., using popular conversion utilities starting from a GIF or JPEG original. Somehow these sneak in every once in a while. They are very large and cannot be effectively compressed. *Avoid them!*

   Instead, you can embed JPEG pictures into a PDF file, a very practical solution for raster graphics. If done properly — i.e., using sufficient resolution! — it will even look acceptable, but never quite as good as vector graphics.

[3] 512×256 pixels $\times 3$ colours = 384 kB; 10% of that is 38 kB.

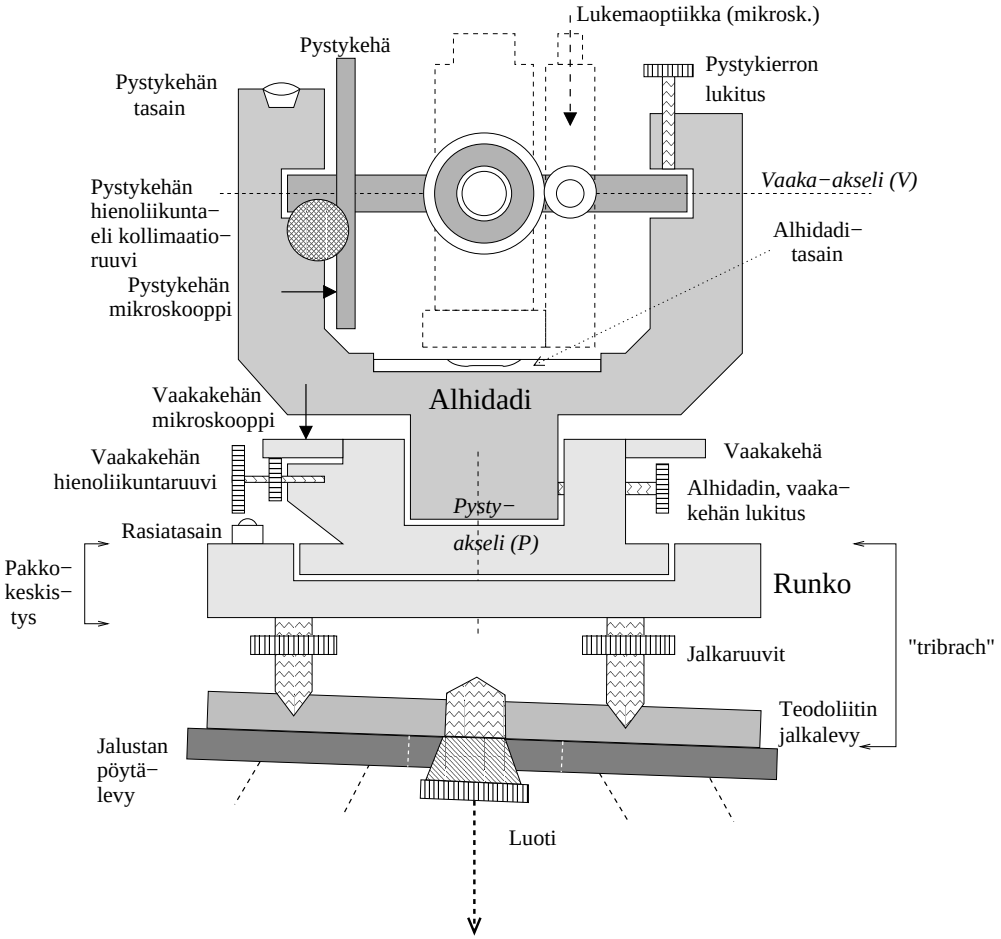

Figure 5: Second example.

EPS:	30 kB	PDF:	17 kB
JPEG:	55 kB	GIF:	32 kB