

Dithering: Making pictures small and aesthetic

clarissa

In this article, we're going to talk about an old trick for pictures that while once borne of necessity is still interesting for reasons related to permacomputing and is honestly just a cool aesthetic anyway: and that's *dithering*.

Dithering, which is a funny sounding word that means tremoring or quivering, is a way to make images use fewer colors while still looking good! The original motivation for this was that older computer monitors were really limited in the number of colors they could display, as opposed to the over 16 million we can display on most monitors today, and we needed to make photos look as good as possible with those limitations! For more on the nature of color and computers, check out another article in this very issue.

The basic idea of dithering is that you use *fewer* colors but then mix the pixels up to give the *impression* of different colors. For example, here's an image (https://commons.wikimedia.org/wiki/File:Black-and-white_photo_made_with_infrared_photography.jpg) that's a monochrome photo with lots of rich gray tones.



And here's the same image dithered so that the only two colors are *black* and *white*.



It still gives the impression of grays though, right? That's because if we intermix white and black pixels next to each other our eye gets the *vibe* of different shades of gray even if on some level you know it's just black and white.

Now, I personally think there's a pretty cool aesthetic to dithering but it still has really practical uses! Why? Because it makes images so much smaller! If you take a photo with a smartphone the image will often be several megabytes in size. If you take a photo with a nice standalone camera the image could be dozens of MB. For our modern storage that might not be huge, but what if you're concerned about permacomputing and want to make a solar-powered image server that you can keep adding photos to for a decade? Then the difference between 10MB and a 100KB dithered version is pretty huge!

But *why* does dithering make images smaller? We've talked about the idea of *compression* in a past article but we'll just briefly review the idea here. Most file formats for images, like jpeg, compress the image by taking redundant data, like say twenty pixels in a row that are all black, and instead of storing twenty pixels worth of data stores the number twenty and then the color black. That's a little oversimplified but gives you the right idea for how image compression works.

What this *means* is that, inherently, if you reduce the number of different colors in an image then it's going to be more compressible!

There's a bunch of different ways to dither, but the basic idea is pretty much always the same: first, you choose a set of colors you want to keep in the final image. In the case above that's just black and white. Then you go through the image pixel by pixel and, for each pixel, you assign the pixel

the final color closest to its original color. Then, and this is the important step, you take the difference between the original color of that pixel and its final color and use that to tweak the other pixels around it that you *haven't* converted yet. This is easiest to image in the black and white case: if you assign a pixel black, make the pixels near it a little lighter. If you assign a pixel white, make the others a little darker. This process gives us that intermixing of white and black pixels to make grey tones!

The main difference between dithering algorithms is *how* they do that process of adjusting other pixels.

If you want to play more with dithering on your own try the Dither It! website (<https://ditherit.com/>) or try implementing your own from linked articles. Either way, try messing around and seeing what cool things you can make!

Further reading:

- <https://surma.dev/things/ditherpunk/>
- <https://shihn.ca/posts/2020/dithering>