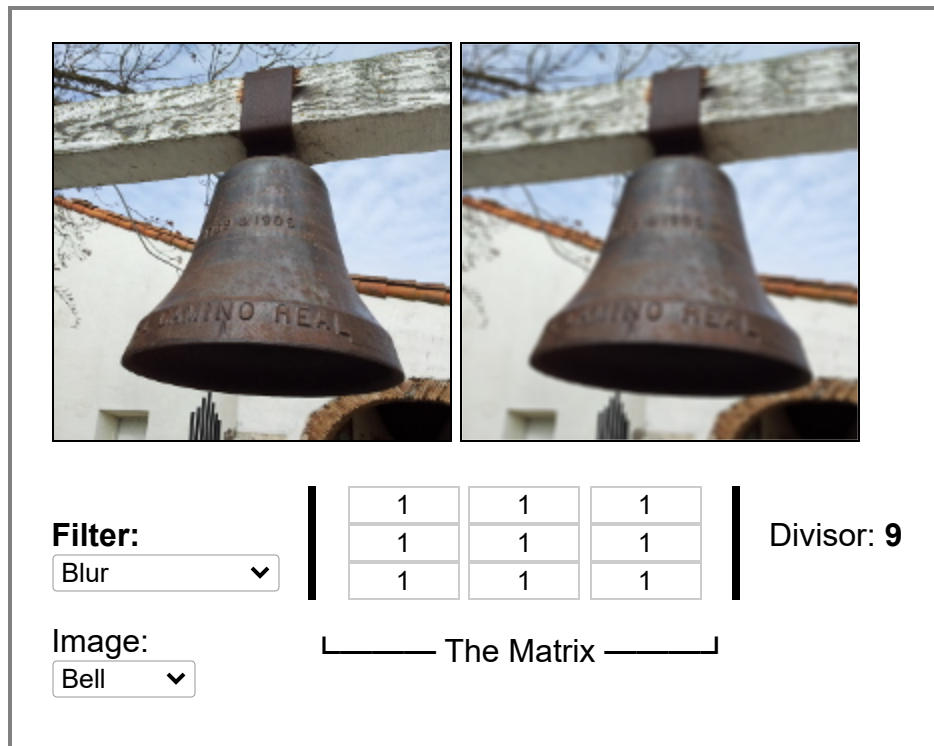


Beej's Bit Bucket

⚡ Tech and Programming Fun

2012-03-24

Image Processing Convolutions



How do image processing filters work, like with blur and sharpen and all that? In this episode, we'll learn the basics!

If you change filters on the app, above, you'll see the values in the matrix change, as well. In fact, you can type in the matrix and put in your own values, too. At this point you might have a question or two.

It is the question that drives us.

Do you know the question?

[What is The Matrix?](#)

In this case, Neo, what we have is a *convolution matrix*.

What we're going to do is generate the destination pixels. To do so, we take data from the corresponding source pixel as well as the source pixel's neighbors. We merge it all together with magic, described below, and the result is the new destination pixel.

In Diagram 1, you can see how we take a 3×3 block of source pixels and turn it into one destination pixel. Also notice that the matrix, above, is a 3×3 matrix. This is no coincidence—each source pixel will be *multiplied* by its corresponding matrix element.

So we multiply these pixel values by the multipliers in the matrix, and then we *add* them all up. Then, lastly, we divide the result by a *divisor* to bring it back down to something sensible. In the case of this demo, the divisor is 9, the same number of elements as are in the matrix.

That is the final destination pixel that we stick in the destination image.

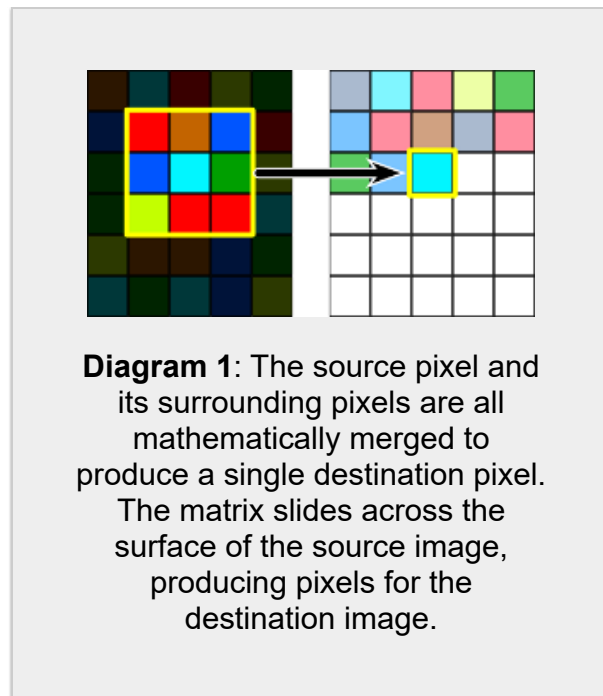
Then we go on to the next pixel. And go through the whole image that way.

The really cool bit is that merely by changing the values in the matrix, we are able to come up with wildly different effects. All the filters in the app, above, run the exact same code—the only difference is the values in the matrix.

Important Safety Tip: don't overwrite source pixels while making the computation. Always leave the source pixels pristine, and write the output to a different destination buffer. If you write to the source buffer, subsequent calculations will be affected, and your result will be flawed. It might look *awesome*, but it will still be incorrect.

Identity, Lighten, and Darken

In the Filter pulldown, above, select "Identity". This is the do-nothing filter. Look at the values in the matrix.



What this filter does is takes the sum of all the surrounding pixels times zero. In other words, the value of all surrounding pixels is just zero. Then we add on 9-times the value of the center pixel. Then we divide it by 9, the divisor. So, basically:

$$D = \frac{9 \times S}{9}$$

$$= S$$

the destination is just the source center pixel. This copies the image, unchanged.

Now look at Lighten. The matrix is just like it is for Identity, except the center value is now 12. So:

$$D = \frac{12 \times S}{9}$$

$$= 1.333 \times S$$

and the destination will be 1.333-times the source center pixel, so brighter.

The Darken filter is the same, except in reverse.

I should note that, yes, this is a very inefficient way to do lighten and darken, since most of the calculations are zero and are just thrown away. It's just included this way to help demonstrate how the matrix affects the result image.

Playtime: change the numbers in the matrix directly and see what comes out.

Blur and Sharpen

Next let's check out the Blur Filter—select it from the pulldown, above, if you haven't already. Then look at the values in the matrix: all **1.0**. What this means is that we're adding all 9 of the (sourcePixel × 1.0), then dividing by **9** (the divisor).

Or, put another way, we're getting the sum of 9 source pixels, and dividing the sum by 9.

Or, put another way, we're *averaging the source pixels for the blur!*

(In a way, that makes a certain kind of sense. If you replace every pixel with the average of the pixels around it, the image is going to

get necessarily "softer", since the hard edge detail is going to be lost in the average.)

So the blur is the straight average. Now select "Sharpen" from the filter list and look at the matrix. We're still doing the same equation as the blur, but we've changed the coefficients. In effect, we're doing a *weighted average*.

And, in fact, we're doing that for all these filters. We just give the surrounding pixels different weights.

More Playtime: What about three rows of "0,3,0"? What about a diagonal of 3s, and zeros elsewhere? How do you modify the numbers to increase the "power" of the Sharpen?

Right about now is a good time to pick up some deferred questions.

Does the matrix have to be 3×3? Not at all. An odd number is good, like 5×5 or 7×7 because it has a center pixel. Keep in mind that processing time increases quite a bit for larger matrices. But you can get better results, too.

Does the divisor have to be 9? Nope. But it's not uncommon for a 3×3, and it helps drive home the weighted-average idea. (Algebraically, it could just be folded into the matrix, and then no divisor would even be necessary to get the same result.)

What does it mean to "multiply a pixel"? Aren't there red, green, and blue channels? Good question! For this demo, I multiplied each channel independently. For a grayscale image, you'd just multiply the gray value, if that makes more sense. You could also do just some channels, or do the intensity, or something else.

Are these sample matrices in the demo the *real* ones? You mean the ones handed down to Moses? No. They're just examples, based on other common examples. Because the demo is just one 3×3 matrix, most of the effects are truly substandard.

Edge Cases

What about the edges? There's no data to the left of a pixel on the left side of the image. So what do you put in the equation for that location?

There are options.

- Assume off-image pixels are black.
- Assume they're white.
- Assume they're some other color.
- Assume they're the color of the nearest pixel on the image.
- Assume the image wraps around, so the row of pixels off the top is the same as the row of pixels on the bottom.
- Assume... something else very complicated.
- Ignore the topmost, bottommost, leftmost, and rightmost lines of pixels, and crop (or mask) them out of the result.

For the demo, it is the last of these options that was taken. In fact, both images in the demo, above are masked by 1-pixel-wide rectangle, clearly visible, to hide the grotesque deformation on the edge of the destination image.

Source and More

- [convolution.js](#)
- [Intro to HTML canvas pixel manipulation](#)

Comments

10 Comments

 Login ▼

G

Join the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS  2

Share

Best

Newest

Oldest

K

ken

7 years ago

buaa increible me ha cambiado my life

0 0 Reply 

K

Ken

10 years ago

So here's a handy next step: combine the use of convolution filters with interpolation or extrapolation. For example, extrapolate from a convolution-blurred image through the original to an unsharp-masked image. Way cool. See <http://graficaobscura.com/i...>

0 0 Reply 

V

Viki

11 years ago

Noob question here. So how is the idea of matrices related to the convolution of signals? or are the similar but not the same idea? I am trying to relate what you have written here with the integral convolution formula Wikipedia (<http://en.wikipedia.org/wik...> offers.

0 0 Reply 

S

Sypher

11 years ago

OMG! The matrix is the IRS!?!

0 0 Reply 

SS

solangi shauban Ali

11 years ago

it is good way of explanation

it is good way of explanation.

0 0 Reply 


AA

Arun Agarwal

— 

11 years ago

I like your explanation , in your discussion I got the answer to my problem

0 0 Reply 

J

Jerson

— 

11 years ago

I like the simple lucid explanation. cheers

0 0 Reply 

M

mike030

— 

12 years ago

Hi Brian,

Excellent stuff. The only problem I found was that you can only use square images ie 200x200. If you use an image say 320x240 it crops the destination image to 240x240.

I am not sure why this happens, but it might be an idea to look at this as most images are 4:3 ratio.

As I say, everything else is excellent

0 0 Reply 



Beej Jorgensen Mod

→ mike030

— 

12 years ago

Hmmm. I must have a bug in there, since it's coded to handle arbitrary widths and heights. I'll have to check it out.

0 0 Reply 

U

ufans

— 

12 years ago

You have a nice way of writing explanation. Keep it up.

Cheers

[Blog](#) ⚡ [Email beej@beej.us](mailto:beej@beej.us) ⚡ [Home page](#)