Image Processing

Mitch Withers, Res. Assoc. Prof., Univ. of Memphis

THE UNIVERSITY OF MEMPHIS
Center for Earthquake Research and Information

Consider the two dimensional function,

$$\phi(x_1, x_2) = \begin{cases} 1, & |x_1|, |x_2| < \frac{1}{2} \\ 0, & |x_1|, |x_2| > \frac{1}{2} \end{cases}$$

A 2-d boxcar.

Consider the two dimensional function,

$$\phi(x_1, x_2) = \begin{cases} 1, & |x_1|, |x_2| < \dfrac{1}{2} \\ 0, & |x_1|, |x_2| > \dfrac{1}{2} \end{cases}$$

A 2-d boxcar.

If $\phi(x_1, x_2)$ is separable, $\phi(x_1, x_2) = h(x_1)h(x_2)$ then,

$$\Phi(f_1, f_2) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x_1)h(x_2)e^{-i2\pi f_1 x_1}e^{-i2\pi f_2 x_2}dx_1 dx_2$$

Consider the two dimensional function,

$$\phi(x_1, x_2) = \begin{cases} 1, & |x_1|, |x_2| < \frac{1}{2} \\ 0, & |x_1|, |x_2| > \frac{1}{2} \end{cases}$$

A 2-d boxcar.

If $\phi(x_1, x_2)$ is separable, $\phi(x_1, x_2) = h(x_1)h(x_2)$ then,

$$\Phi(f_1, f_2) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x_1)h(x_2)e^{-i2\pi f_1 x_1}e^{-i2\pi f_2 x_2}dx_1 dx_2$$

$$= \int_{-\infty}^{\infty} h(x_1)e^{-i2\pi f_1 x_1}dx_1 \int_{-\infty}^{\infty} h(x_2)e^{-i2\pi f_2 x_2}dx_2$$

Consider the two dimensional function,

$$\phi(x_1, x_2) = \begin{cases} 1, & |x_1|, |x_2| < \frac{1}{2} \\ 0, & |x_1|, |x_2| > \frac{1}{2} \end{cases}$$

A 2-d boxcar.

If $\phi(x_1, x_2)$ is separable, $\phi(x_1, x_2) = h(x_1)h(x_2)$ then,

$$\Phi(f_1, f_2) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x_1)h(x_2)e^{-i2\pi f_1 x_1}e^{-i2\pi f_2 x_2}dx_1 dx_2$$

$$= \int_{-\infty}^{\infty} h(x_1)e^{-i2\pi f_1 x_1}dx_1 \int_{-\infty}^{\infty} h(x_2)e^{-i2\pi f_2 x_2}dx_2$$

$$= \int_{-1/2}^{1/2} e^{-i2\pi f_1 x_1}dx_1 \int_{-1/2}^{1/2} e^{-i2\pi f_2 x_2}dx_2$$

Consider the two dimensional function, $\phi(x_1,x_2) = \begin{cases} 1, & |x_1|,|x_2| < \frac{1}{2} \\ 0, & |x_1|,|x_2| > \frac{1}{2} \end{cases}$ A 2-d boxcar.

If $\phi(x_1,x_2)$ is separable, $\phi(x_1,x_2) = h(x_1)h(x_2)$ then,

$$\Phi(f_1,f_2) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x_1)h(x_2)e^{-i2\pi f_1 x_1}e^{-i2\pi f_2 x_2}dx_1dx_2$$

$$= \int_{-\infty}^{\infty} h(x_1)e^{-i2\pi f_1 x_1}dx_1 \int_{-\infty}^{\infty} h(x_2)e^{-i2\pi f_2 x_2}dx_2$$

$$= \int_{-1/2}^{1/2} e^{-i2\pi f_1 x_1}dx_1 \int_{-1/2}^{1/2} e^{-i2\pi f_2 x_2}dx_2$$

$$= 2\int_0^{1/2} \cos(2\pi f_1 x_1)dx_1 \cdot 2\int_0^{1/2} \cos(2\pi f_2 x_2)dx_2$$

Consider the two dimensional function,

$$\phi(x_1, x_2) = \begin{cases} 1, & |x_1|, |x_2| < \dfrac{1}{2} \\ 0, & |x_1|, |x_2| > \dfrac{1}{2} \end{cases}$$

A 2-d boxcar.

If $\phi(x_1, x_2)$ is separable, $\phi(x_1, x_2) = h(x_1)h(x_2)$ then,

$$\Phi(f_1, f_2) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x_1)h(x_2)e^{-i2\pi f_1 x_1}e^{-i2\pi f_2 x_2}dx_1 dx_2$$

$$= \int_{-\infty}^{\infty} h(x_1)e^{-i2\pi f_1 x_1}dx_1 \int_{-\infty}^{\infty} h(x_2)e^{-i2\pi f_2 x_2}dx_2$$

$$= \int_{-1/2}^{1/2} e^{-i2\pi f_1 x_1}dx_1 \int_{-1/2}^{1/2} e^{-i2\pi f_2 x_2}dx_2$$

$$= 2\int_{0}^{1/2} \cos(2\pi f_1 x_1)dx_1 \cdot 2\int_{0}^{1/2} \cos(2\pi f_2 x_2)dx_2$$

$$= \frac{\sin(\pi f_1)}{\pi f_1} \cdot \frac{\sin(\pi f_2)}{\pi f_2}$$

THE UNIVERSITY OF MEMPHIS. Center for Earthquake Research and Information

When calculating the FT with the maltab fft command in 1-D we operate on a single vector,

$$\phi(x) = [x_0\; x_1\; x_2 \cdots x_{N-1}]$$

When calculating the FT with the maltab fft command in 1-D we operate on a single vector,

$$\phi(x) = [x_0\ x_1\ x_2 \cdots x_{N-1}]$$

But in two dimensions we have a matrix,

$$\phi(m,n) = \begin{bmatrix} \phi_{0,0} & \cdots & \phi_{m-1,0} \\ \vdots & \ddots & \vdots \\ \phi_{0,n-1} & \cdots & \phi_{m-1,n-1} \end{bmatrix}$$

When calculating the FT with the maltab fft command in 1-D we operate on a single vector,

$$\phi(x) = [x_0\ x_1\ x_2 \cdots x_{N-1}]$$

But in two dimensions we have a matrix,

$$\phi(m,n) = \begin{bmatrix} \phi_{0,0} & \cdots & \phi_{m-1,0} \\ \vdots & \ddots & \vdots \\ \phi_{0,n-1} & \cdots & \phi_{m-1,n-1} \end{bmatrix}$$

So in practice, we can calculate the FT on each row (or column) to form an intermediate matrix, then calculate the FT on each column (or row) for the final 2-D FT. So we have to calculate $M \cdot N$ FTs.

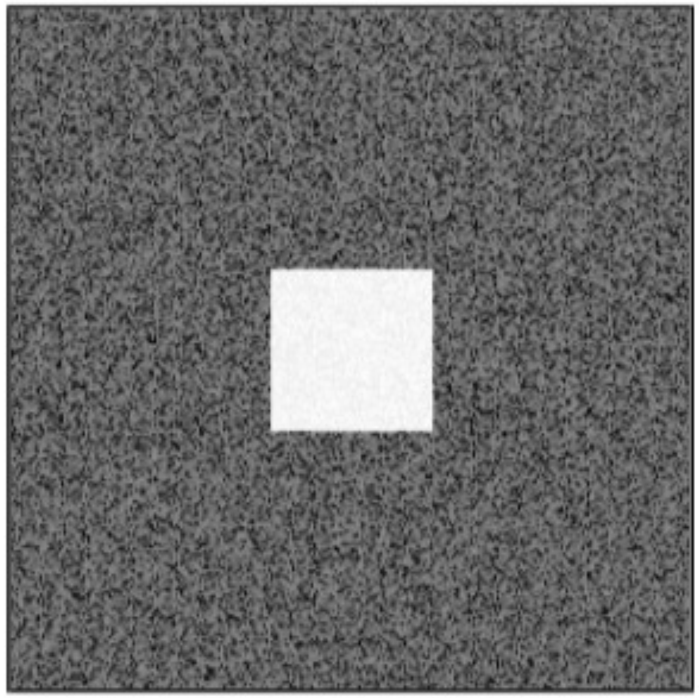
Matlab has a 2-d fft command: fft2.

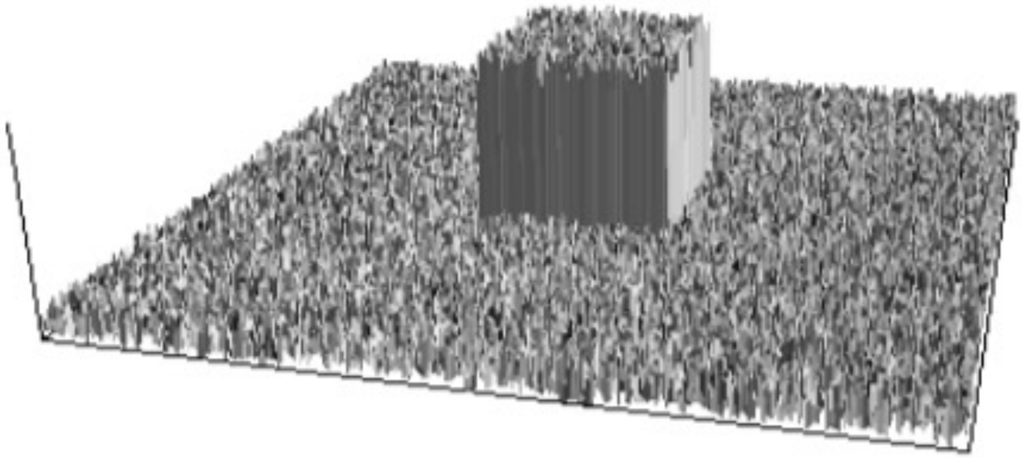
The matlab fftshift, also works in 2-d.

Run matlab scripts BoxCar2.m then BoxCar2b.m

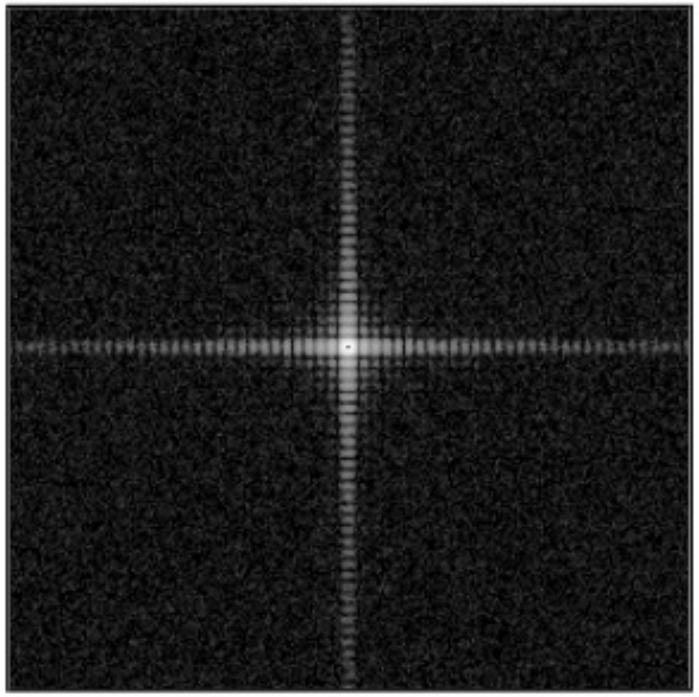
Image Filtering in the frequency domain from Paul Bourke
http://paulbourke.net/miscellaneous/imagefilter/
Last accessed July 15, 2020

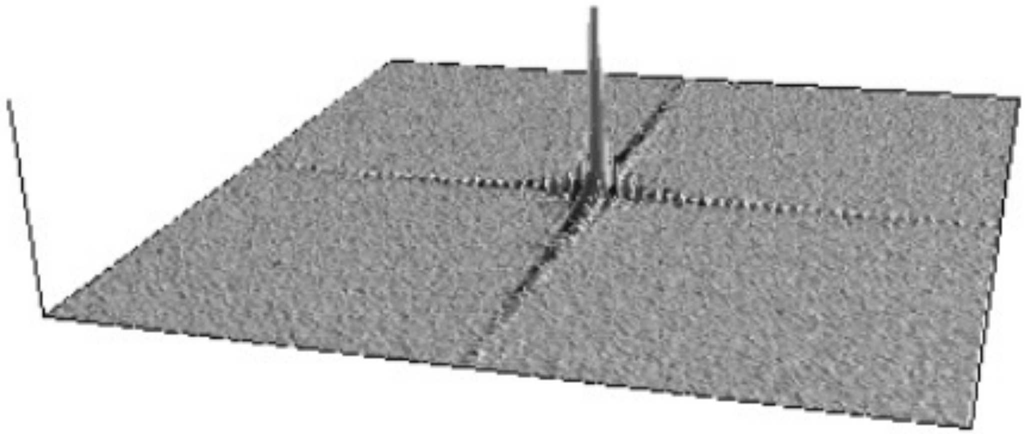
We start with a 2-d boxcar in the spatial domain, FT and apply various low, high and bandpass filters. We show both map and 3-d views for clarity. A small amount of noise is added.





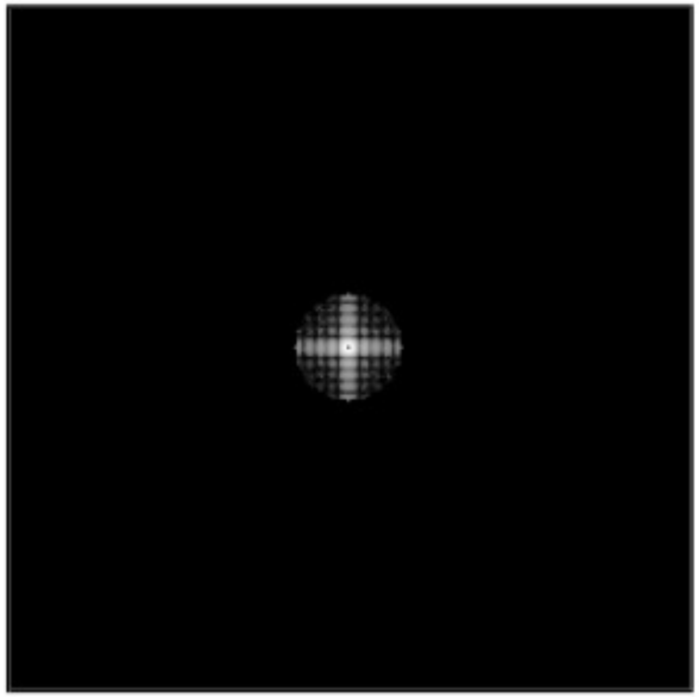
The FT of the 2-d boxcar is a 2-d sync function.

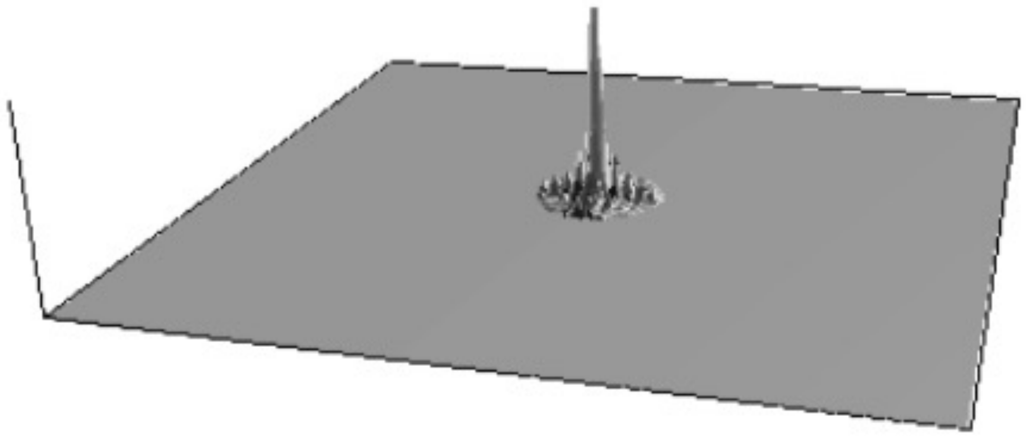




THE UNIVERSITY OF MEMPHIS
Center for Earthquake Research and Information

We can apply an ideal low-pass filter, which is a cylinder in 2-d frequency, by direct multiplication of the amplitude spectrum. Note 0 frequency (aka DC) is at the center of the image.

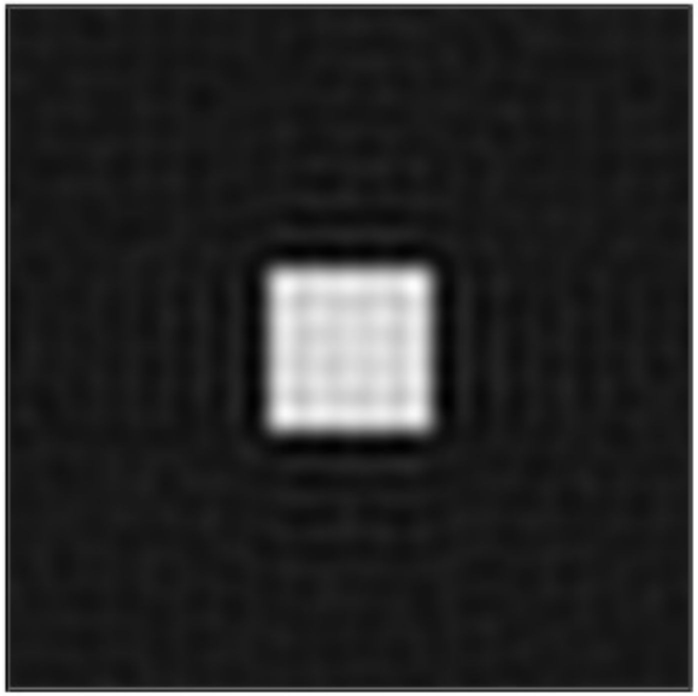


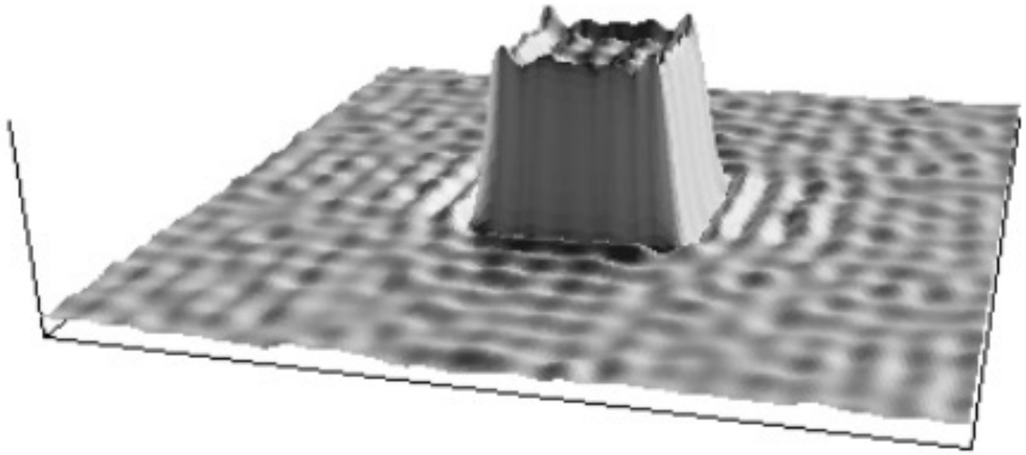


Transform the filtered signal back into the spatial domain.

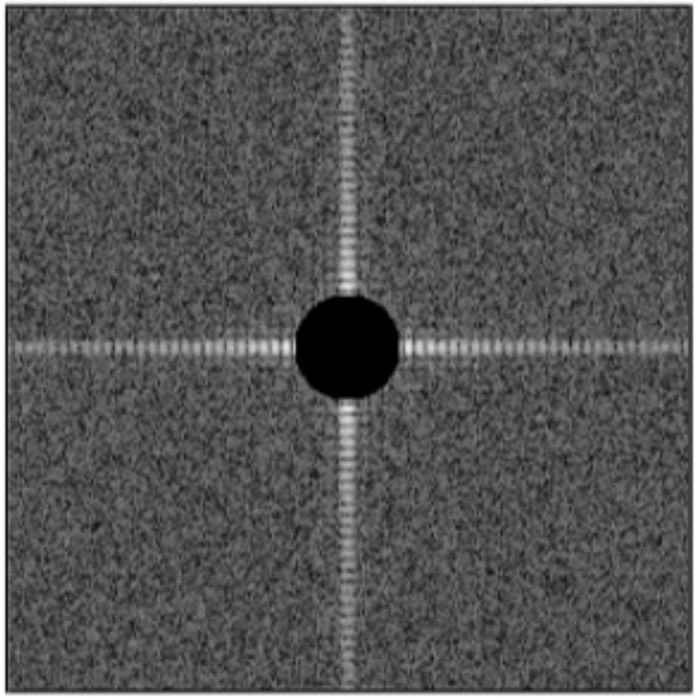
The sharp edge of the boxcar is blurred due to the loss of high frequency information.

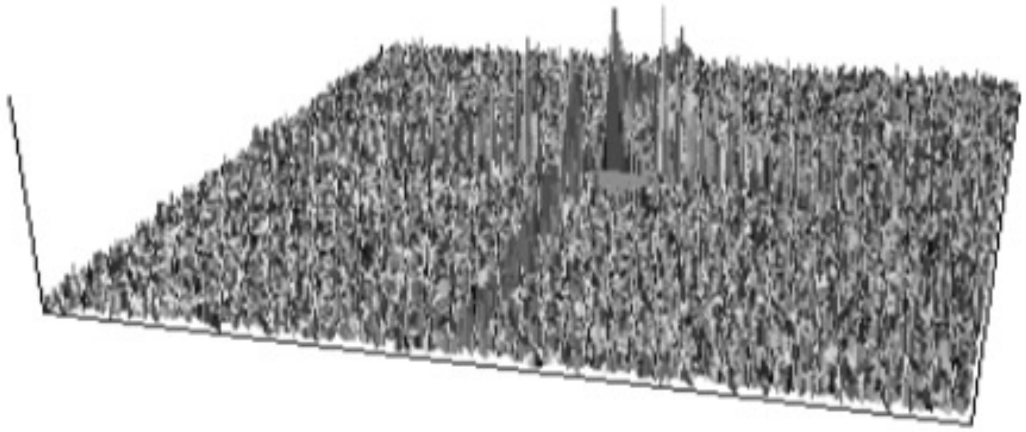
We've also introduced a lot of ripple due to the shape of our filter.





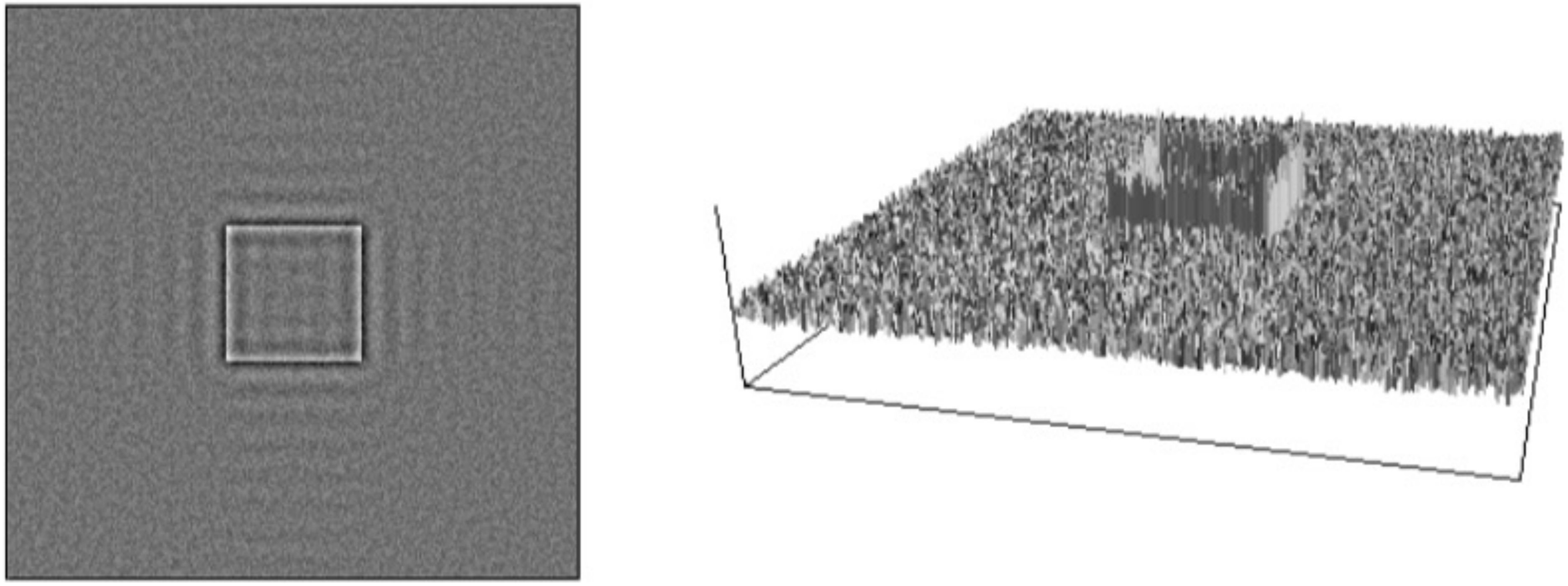
Back in the frequency domain, let's apply an ideal high pass filter to the amplitude spectrum.



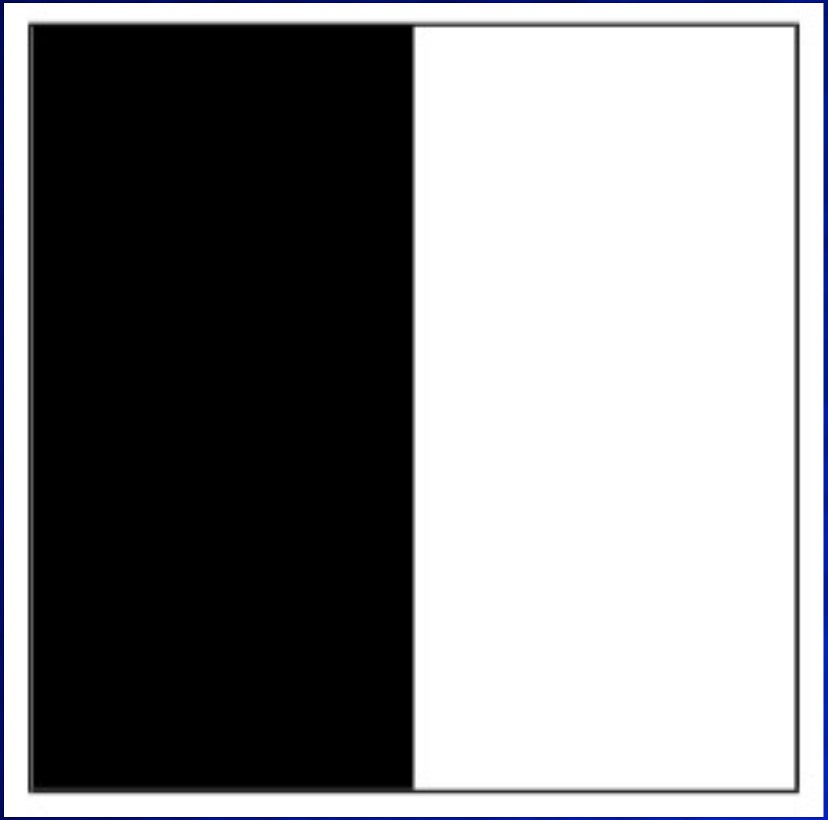


THE UNIVERSITY OF MEMPHIS. Center for Earthquake Research and Information

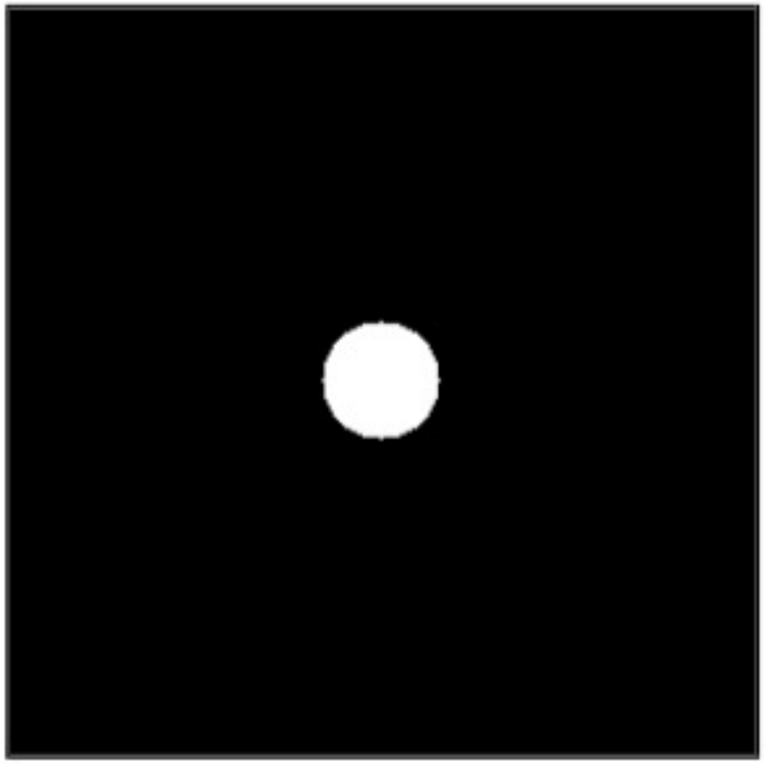
Now we’ve mostly preserved the sharp edge but lost the low frequencies necessary to construct the flat part of the boxcar.

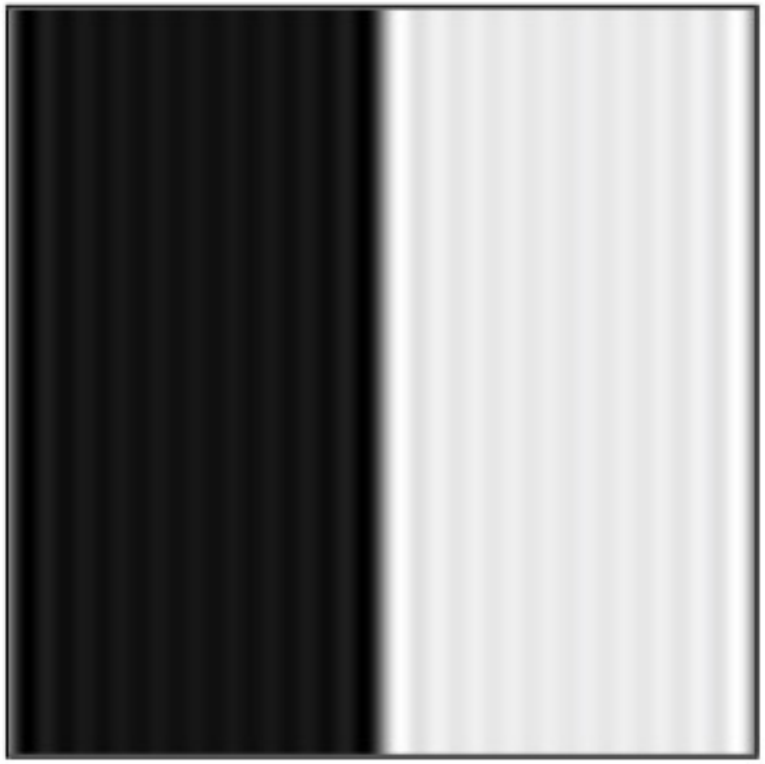


Next we use a 2-d step function as our spatial domain signal.



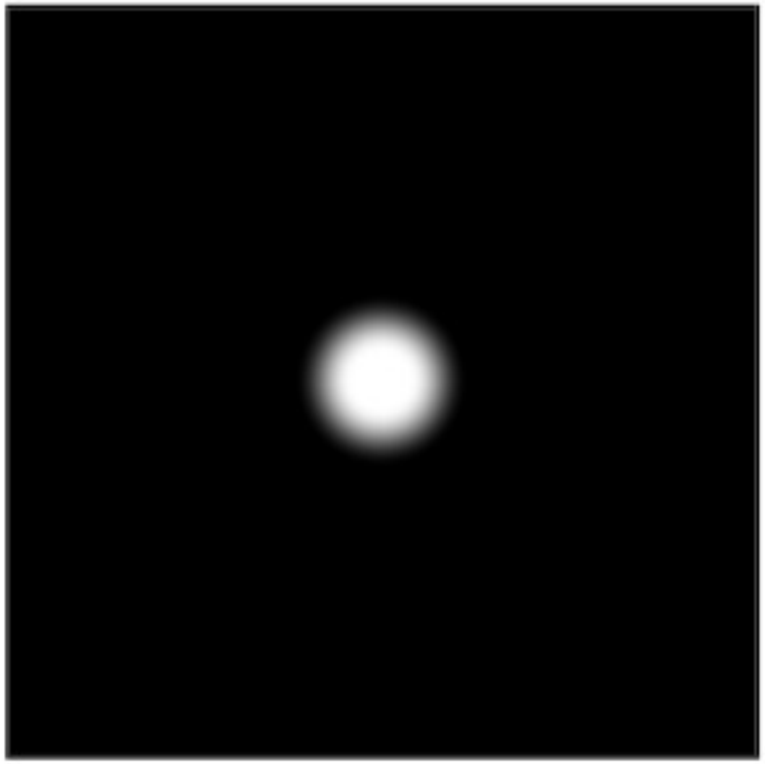
If we apply an ideal low pass filter, we blur the sharp transition of the step function and introduce ripple in the spatial domain because of the sharp transition of the filter. Recall multiplying by a boxcar in frequency is the same as convolving by a sync in space (or time).

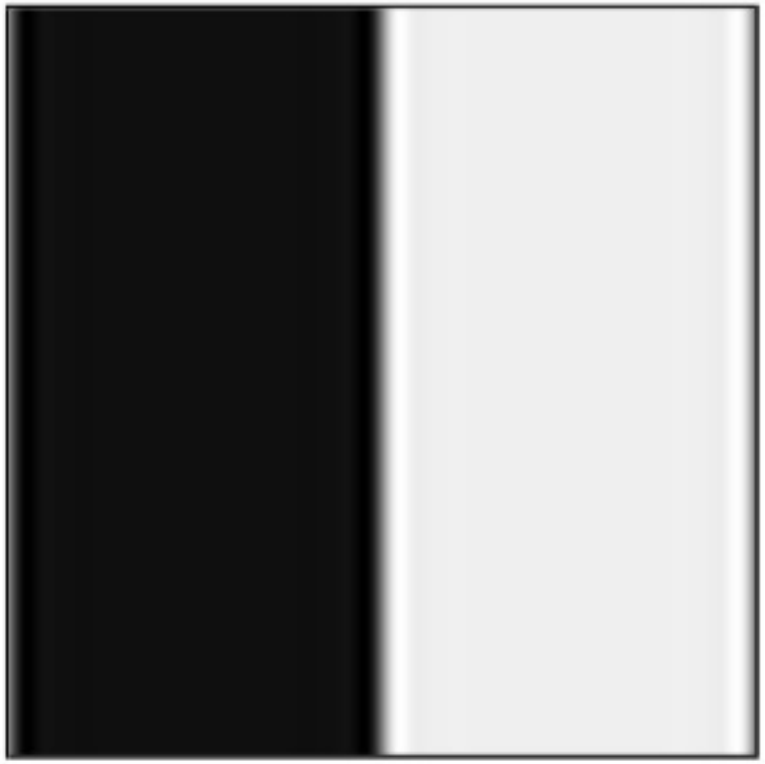




THE UNIVERSITY OF MEMPHIS. Center for Earthquake Research and Information

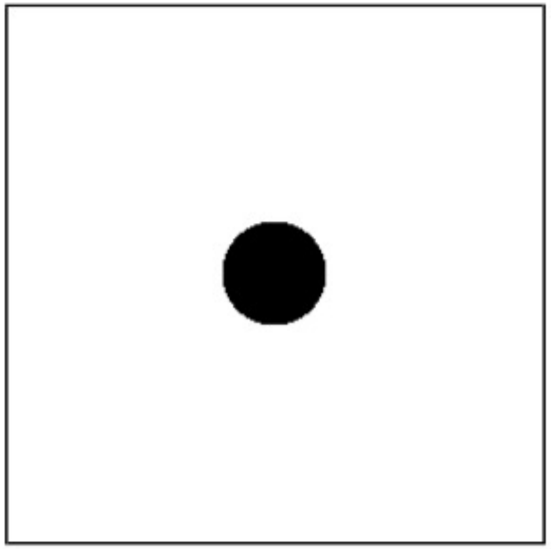
We can smooth the edge of the lowpass to reduce the ripple in the spatial domain at the expense of a wider transition band.

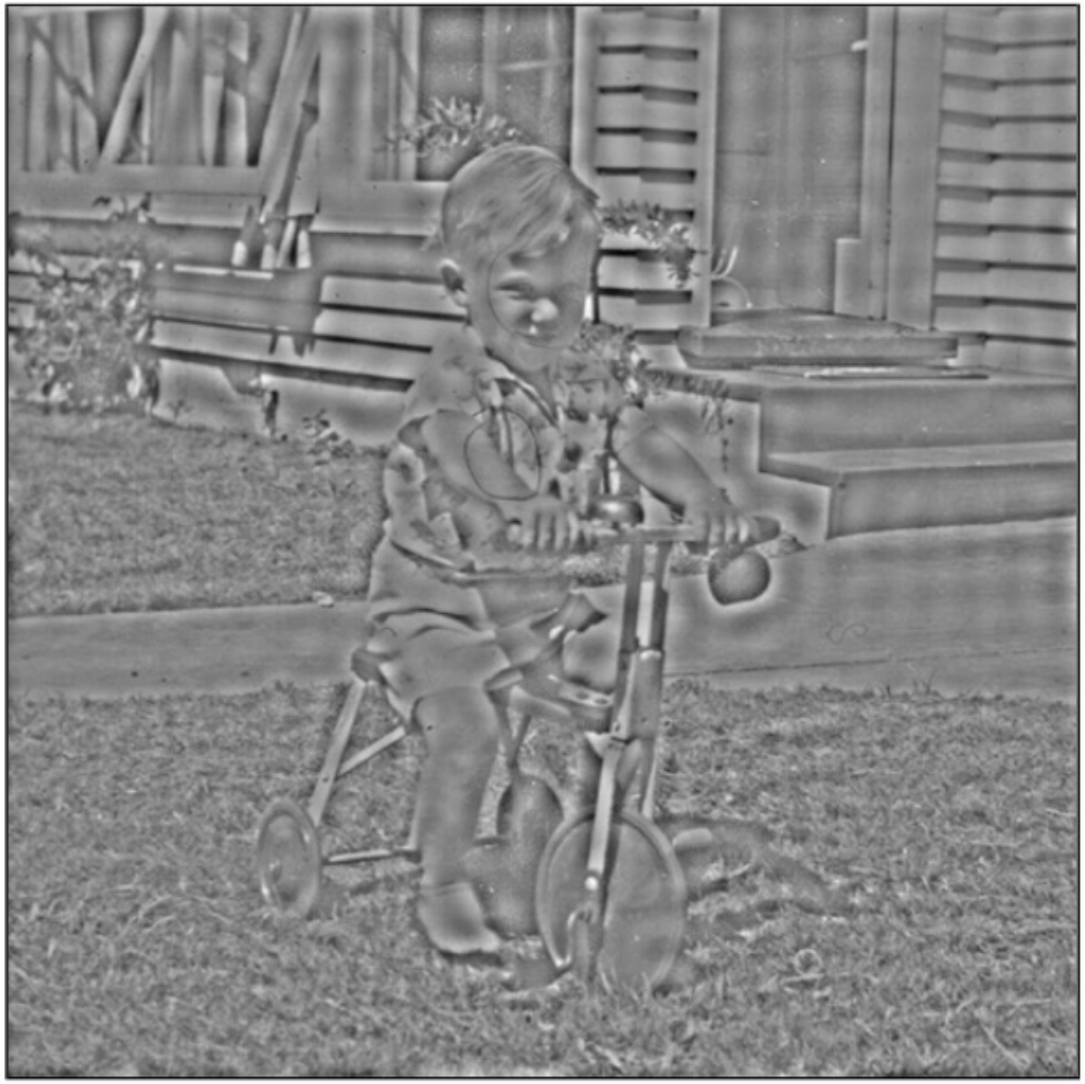




The can apply all the filtering techniques we discussed to a picture (a very old one in this case) which, once it's been digitized (e.g. scanned into a computer) is just a matrix of pixels.

Here is the sharp high pass filter. Note the ringing introduced especially visible on the sidewalk.





The ringing can be cleaned up by smoothing the edge of the filter.

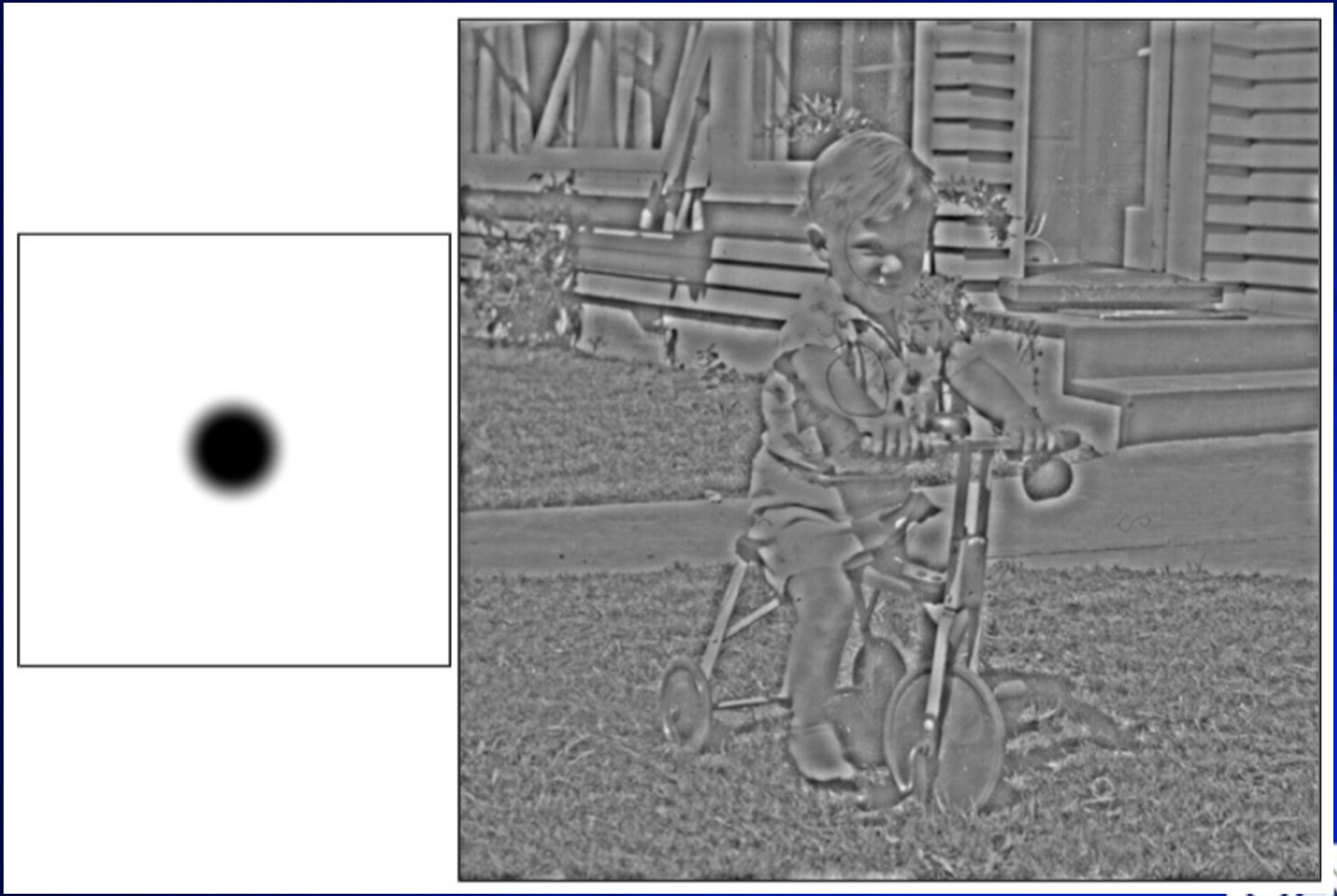


Image Filtering from Lode Vandeveene
https://lodev.org/cgtutor/filtering.html
Last accessed July 15, 2020

Filtering may sometimes be more intuitive in the spatial domain. In this tutorial, convolution with a relatively small matrix (3x3, 5x5, or 7x7 pixels) is used to produce different effects on the image. What can't be done is CSI style "enhance" (you can't add data where it doesn't exist).

Image Filtering from Lode Vandeveene
https://lodev.org/cgtutor/filtering.html
Last accessed July 15, 2020

Filtering may sometimes be more intuitive in the spatial domain. In this tutorial, convolution with a relatively small matrix (3x3, 5x5, or 7x7 pixels) is used to produce different effects on the image. What can't be done is CSI style "enhance" (you can't add data where it doesn't exist).

If we convolve an image with this matrix, what will be the result?

$$\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}$$

THE UNIVERSITY OF MEMPHIS.
Center for Earthquake Research and Information

Image Filtering from Lode Vandeveene
https://lodev.org/cgtutor/filtering.html
Last accessed July 15, 2020

Filtering may sometimes be more intuitive in the spatial domain. In this tutorial, convolution with a relatively small matrix (3x3, 5x5, or 7x7 pixels) is used to produce different effects on the image. What can't be done is CSI style "enhance" (you can't add data where it doesn't exist).

If we convolve an image with this matrix, what will be the result?

$$\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}$$

Recall the sifting property of the delta function,

$$\int_a^b f(t)\delta(t-t_0)dt = f(t_0), \qquad where\ a \le t_0 \le b$$

THE UNIVERSITY OF MEMPHIS
Center for Earthquake Research and Information

Image Filtering from Lode Vandeveene
https://lodev.org/cgtutor/filtering.html
Last accessed July 15, 2020

Filtering may sometimes be more intuitive in the spatial domain. In this tutorial, convolution with a relatively small matrix (3x3, 5x5, or 7x7 pixels) is used to produce different effects on the image. What can't be done is CSI style "enhance" (you can't add data where it doesn't exist).

If we convolve an image with this matrix, what will be the result?

$$\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}$$

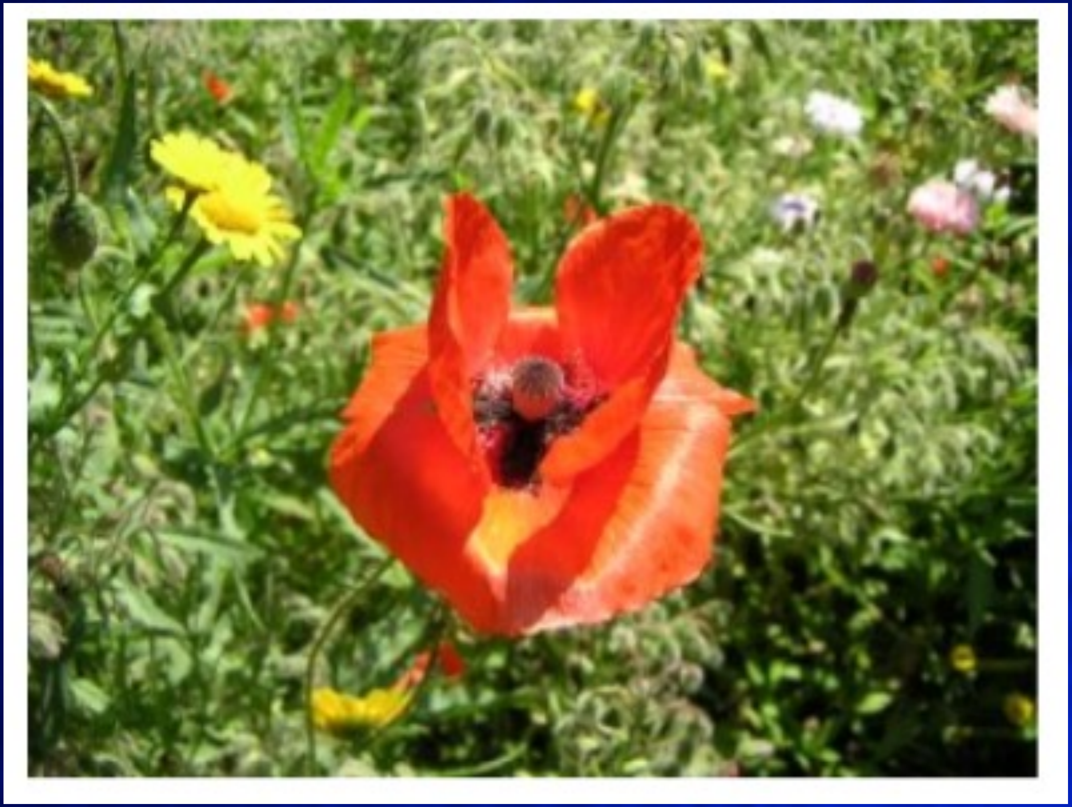
Recall the sifting property of the delta function,

$$\int_a^b f(t)\delta(t-t_0)dt = f(t_0), \qquad where\ a \le t_0 \le b$$

As we move the filter matrix around the image so that the center of the matrix is in turn centered on each pixel of the image, each individual multiply and sum just returns the pixel at the center of the matrix. We get the same image out as we put in.

THE UNIVERSITY OF MEMPHIS. Center for Earthquake Research and Information

The original image we'll be working with.



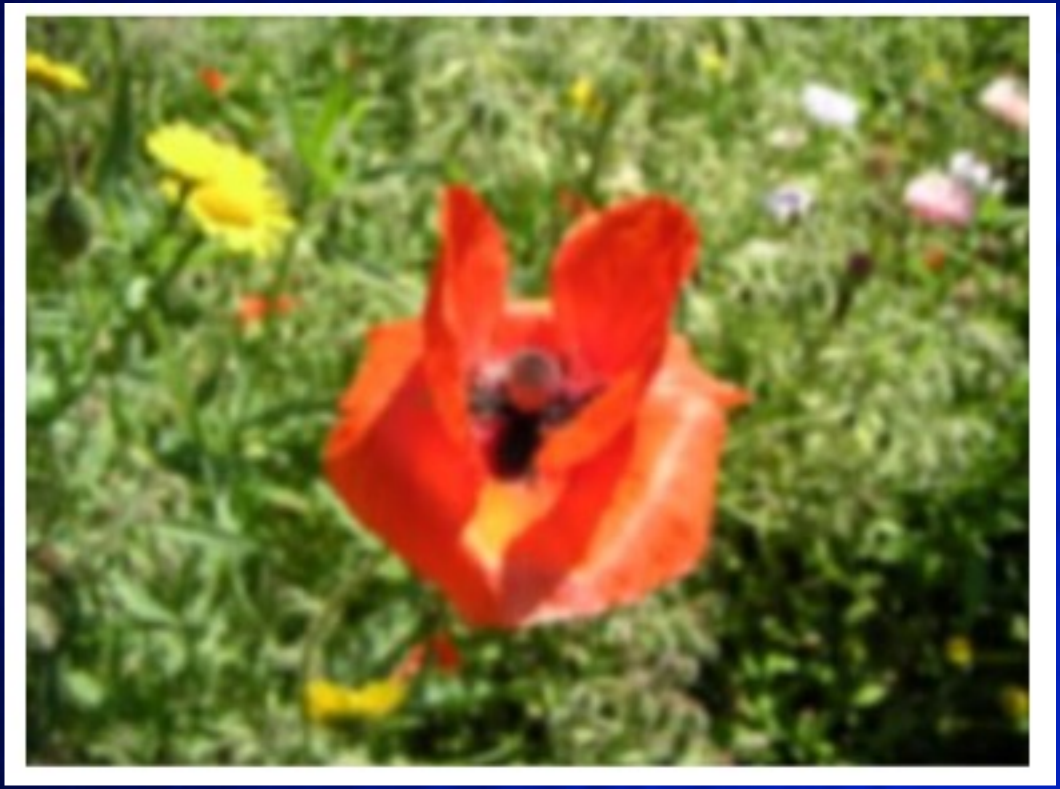
We can blur the image by finding the average of each pixel with it’s neighbor implemented by this filter matrix

$$\begin{matrix} 0.0 & 0.2 & 0.0 \\ 0.2 & 0.2 & 0.2 \\ 0.0 & 0.2 & 0.0 \end{matrix}$$

With such a small filter matrix, the blur is subtle.

We can blur more with a bigger matrix. We’ll need to take care to normalize by the number of 1’s present in the matrix to avoid brightening the image.

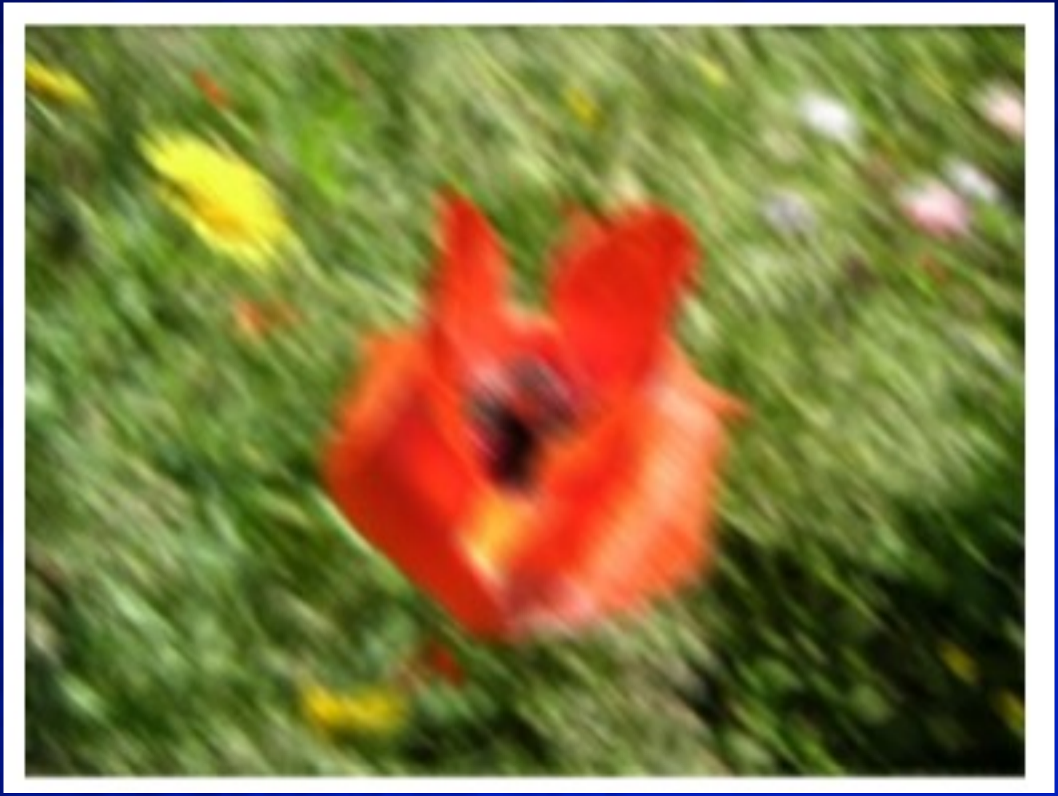
$$\begin{matrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{matrix}$$



THE UNIVERSITY OF MEMPHIS
Center for Earthquake Research and Information

If we blur in a specific direction we can make an image that looks like it’s in motion.

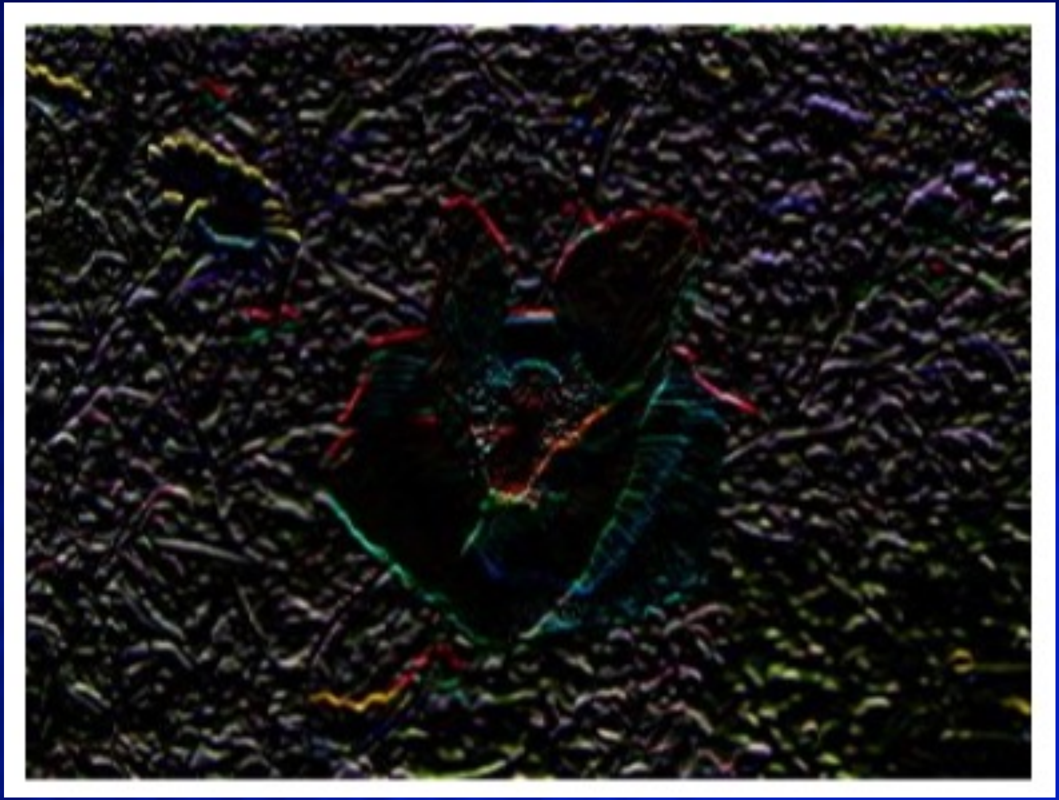
$$\begin{matrix} \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \end{matrix}$$



THE UNIVERSITY OF MEMPHIS. Center for Earthquake Research and Information

We can detect edges in the image by simulating the derivative in the filter and finding the difference between a pixel and it's near neighbors. If we do this vertically we detect horizontal edges.

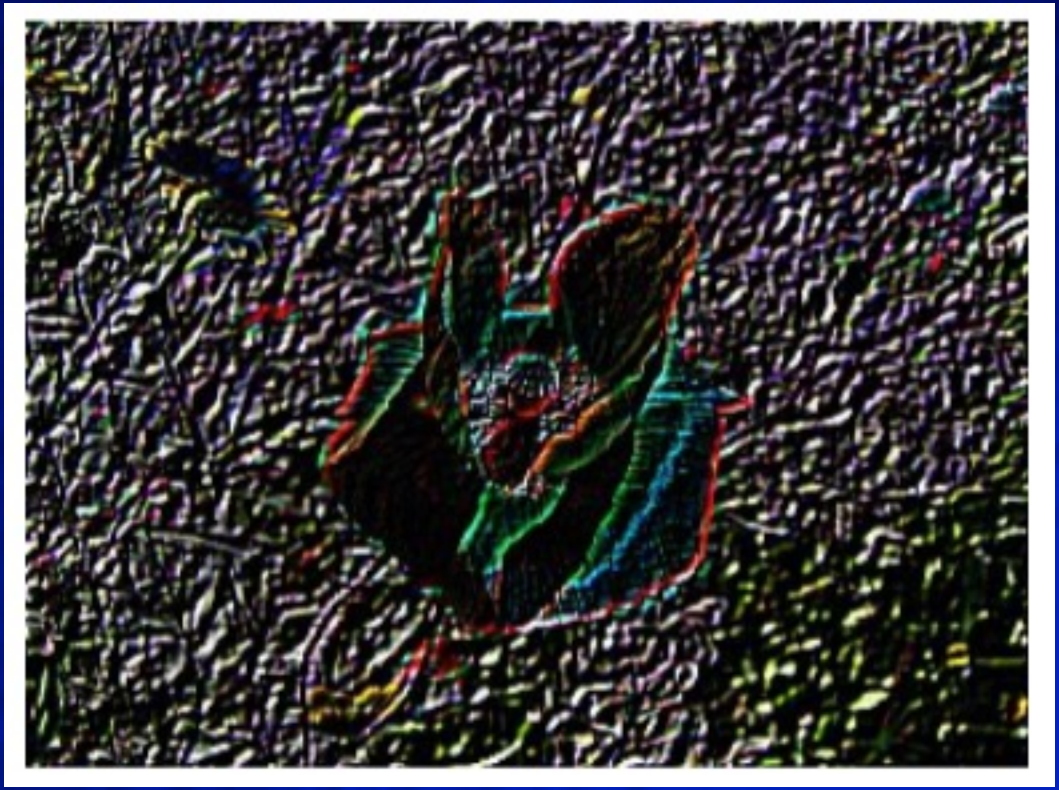
$$\begin{matrix} 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$



The sum of the elements in the filter is 0 which results in a darkened image with only the edges colored.

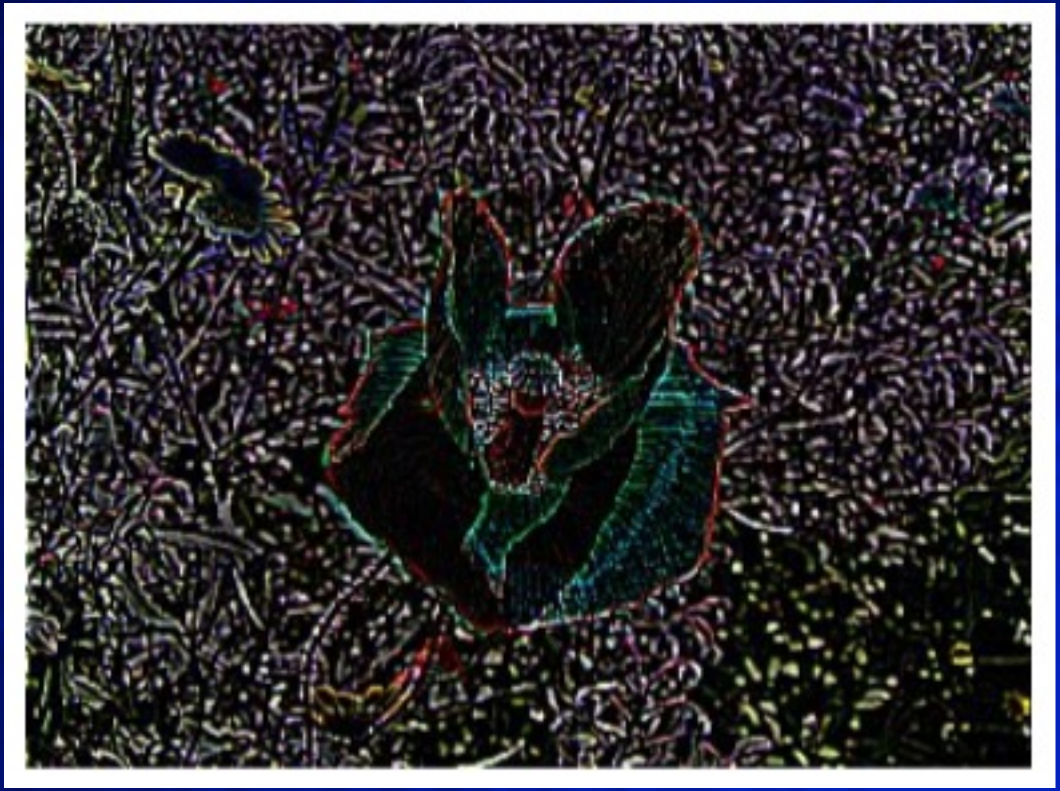
This filter finds edges diagonally.

$$\begin{matrix} -1 & 0 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 & 0 \\ 0 & 0 & 6 & 0 & 0 \\ 0 & 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 0 & -1 \end{matrix}$$



An omni directional edge detector.

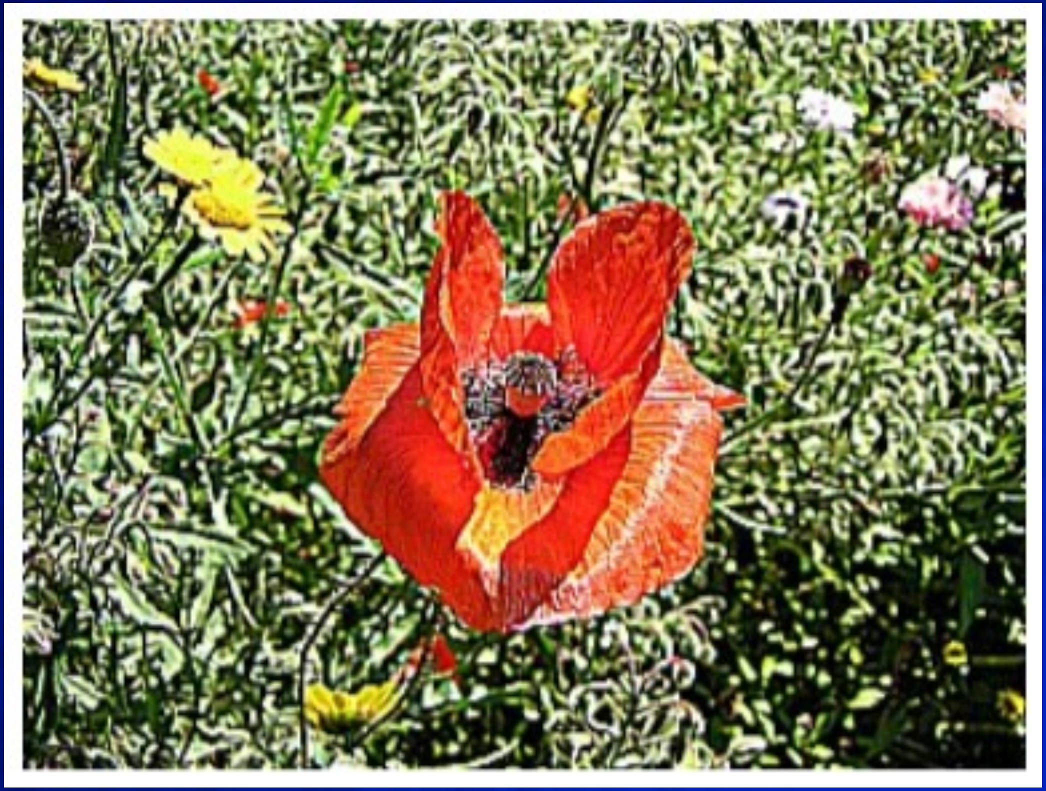
$$\begin{matrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{matrix}$$



THE UNIVERSITY OF MEMPHIS. Center for Earthquake Research and Information

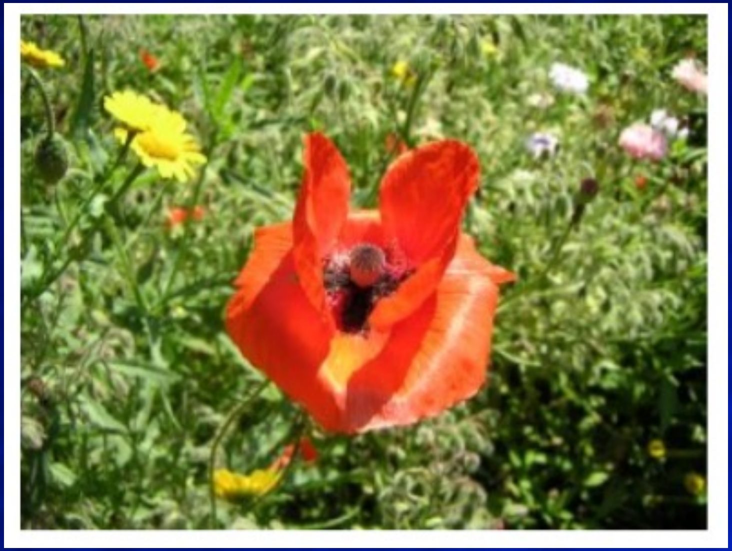
To sharpen an image add the original image to the edge detection filtered image. So instead of 8 at the center of the 3x3 matrix to make it sum to 0, set the center to 9 which adds in a copy of the original using the sifting property.

$$\begin{matrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{matrix}$$

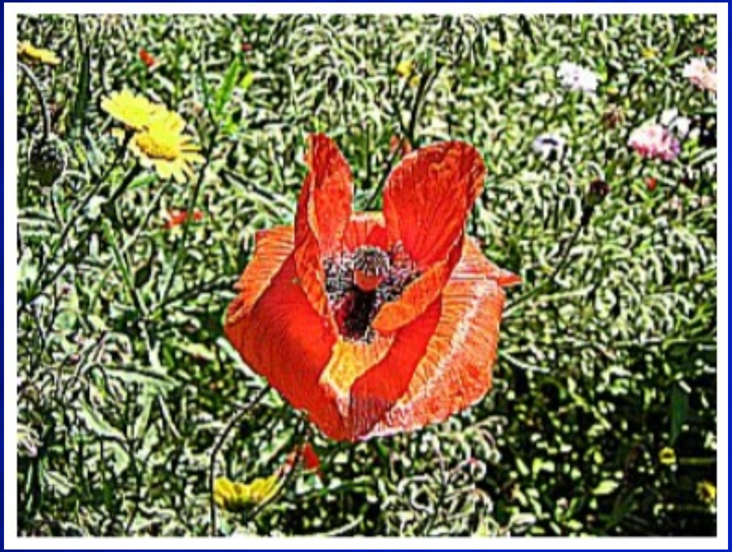


To sharpen an image add the original image to the edge detection filtered image. So instead of 8 at the center of the 3x3 matrix to make it sum to 0, set the center to 9 which adds in a copy of the original using the sifting property.

$$\begin{matrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{matrix}$$



Original

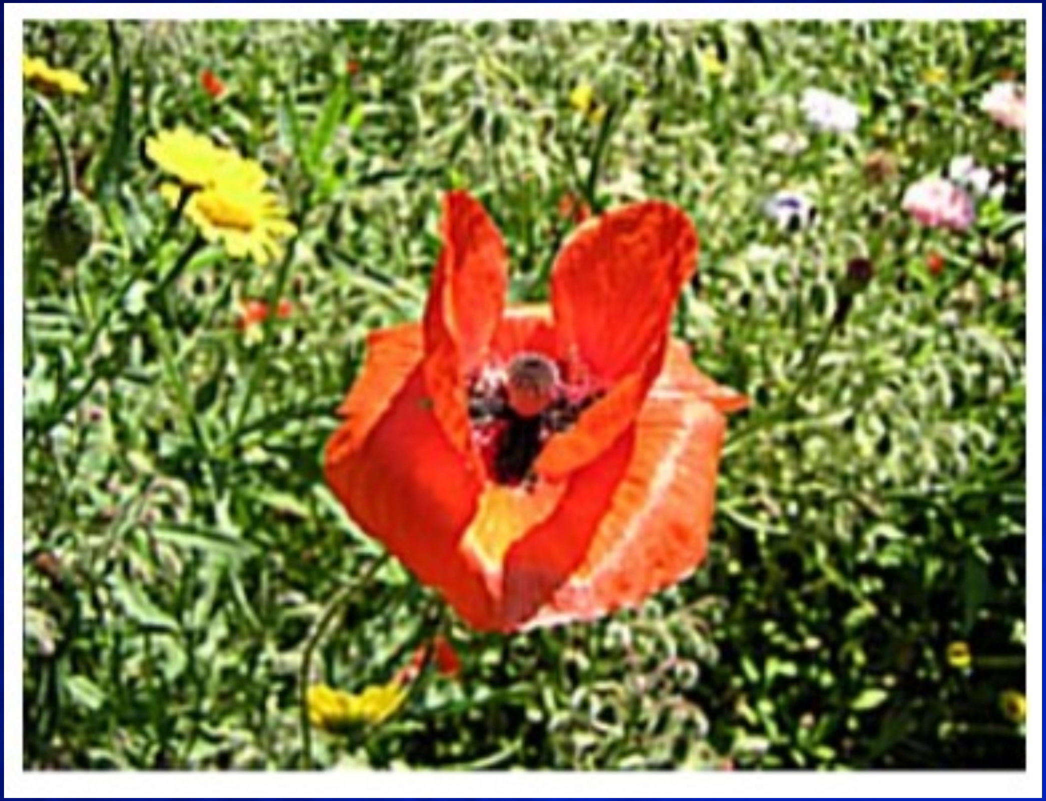


Sharpened

THE UNIVERSITY OF MEMPHIS
Center for Earthquake Research and Information

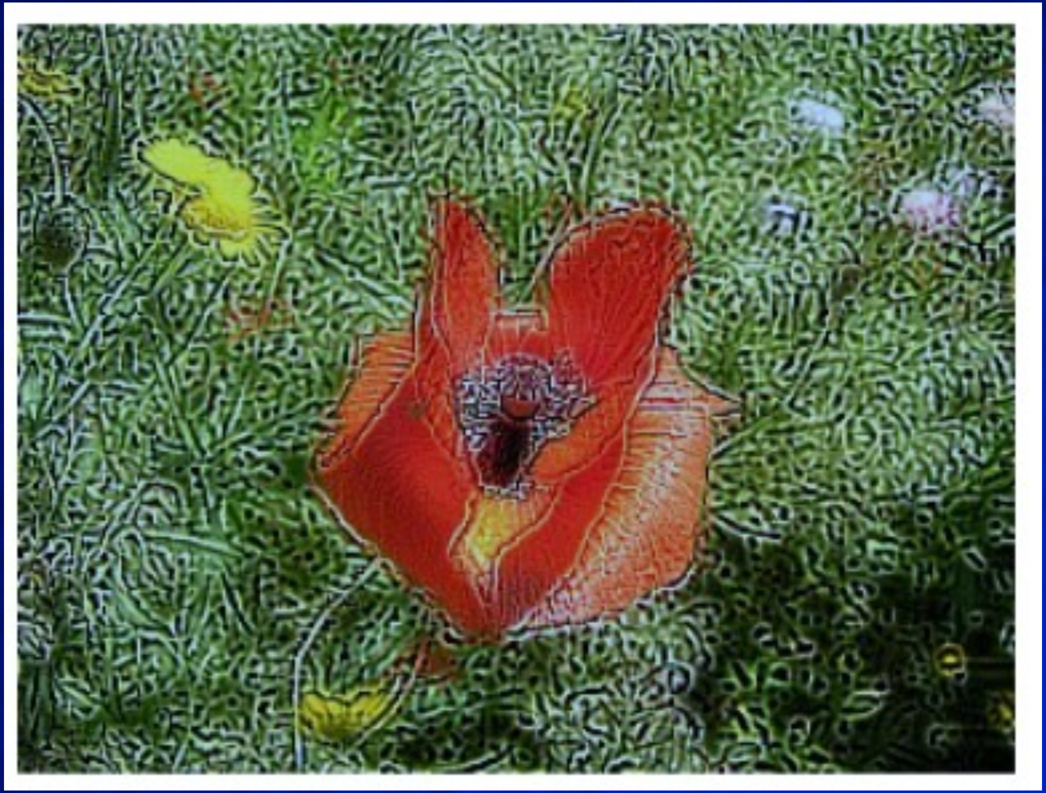
A more subtle sharpening filter

-1	-1	-1	-1	-1
-1	2	2	2	2
-1	2	8	2	-1
-1	2	2	2	-1
-1	-1	-1	-1	-1



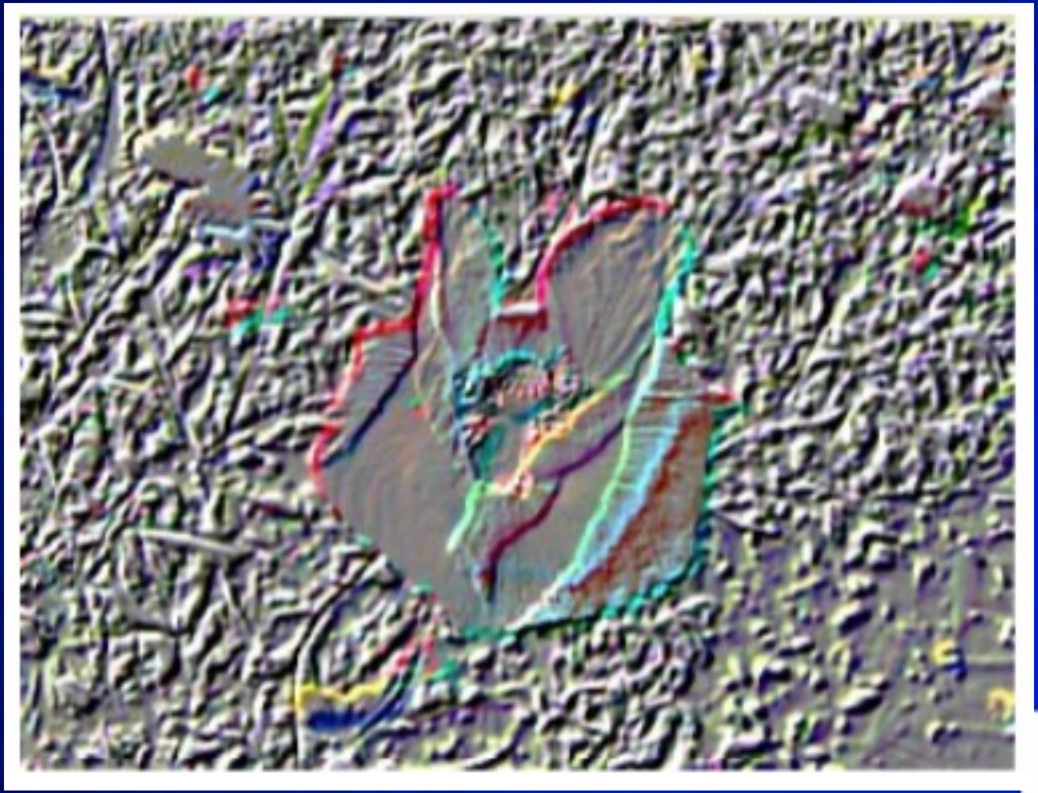
Here’s an filter that can excessively enhance edges by essentially subtracting the image from the edges.

$$\begin{matrix} 1 & 1 & 1 \\ 1 & -7 & 1 \\ 1 & 1 & 1 \end{matrix}$$



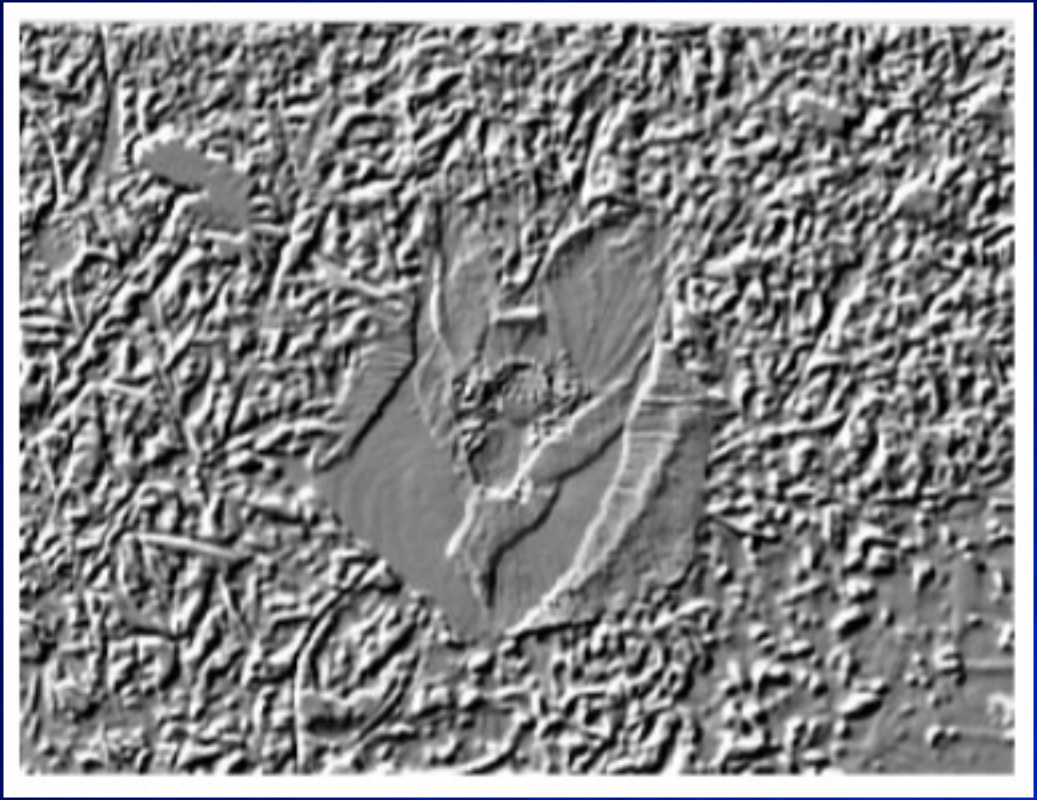
For the emboss filter, we quote Vandevenne directly, “An emboss filter gives a 3D shadow effect to the image, the result is very useful for a bumpmap of the image. It can be achieved by taking a pixel on one side of the center, and subtracting one of the other side from it. Pixels can get either a positive or a negative result. To use the negative pixels as shadow and positive ones as light, for a bumpmap, a bias of 128 is added to the image. Now, most parts of the image will be gray, and the sides will be either dark gray/black or bright gray/white.”

$$\begin{array}{ccc} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{array}$$



THE UNIVERSITY OF MEMPHIS
Center for Earthquake Research and Information

Same filter but with the output converted from RGB to grayscale



There are more image processing techniques in the original document referenced at the beginning of this file along with examples of c++ code to implement the filters shown.