1

a

$W2 = ((W1 - Fw + 2Pw) / Sw) + 1$

$H2 = ((H1 - Fh + 2Ph) / Sh) + 1$

$W2 = W1$ and $H2 = H1$

$Sw = Sh = 1$

$Fw = Fh = 5$

$W2 = ((W1 - 5 + 2Pw) / 1) + 1$

$W1 = (W1 - 5 + 2Pw) + 1$

$2Pw = W1 - W1 + 5 - 1$

$Pw = (5 - 1) / 2 = 2$

$H2 = ((H1 - Fh + 2Ph) / Sh) + 1$

$H1 = (H1 - 5 + 2Ph) + 1$

$2Ph = H1 - H1 + 5 - 1$

$Ph = (5 - 1) / 2 = 2$

b

$S = 1$

$F =$ odd

$W2 = H2 = 504$

$W2 = ((W1 - Fw + 2Pw) / Sw) + 1$

$504 = ((512 - Fw + 0) / 1) + 1$

504 = (512 - Fw) + 1

Fw = 512 - 504 + 1 = 9

H2 = ((H1 - Fh + 2Ph) / Sh) + 1

504 = ((512 - Fh + 0) / 1) + 1

504 = (512 - Fh) + 1

Fw = 512 - 504 + 1 = 9

Spatial dimensions of kernel = (9 x 9)

c

W2 = ((W1 - Fw + 2Pw) / Sw) + 1

W2 = ((504 - 2 + 0) / 2) + 1

W2 = 252

H2 = ((H1 - Fh + 2Ph) / Sh) + 1

H2 = ((504 - 2 + 0) / 2) + 1

H2 = 252

Spatial dimensions of the pooled feature maps = (252 x 252)

d

W2 = ((W1 - Fw + 2Pw) / Sw) + 1

W2 = ((252 - 3 + 0) / 1) + 1

W2 = 250

H2 = ((H1 - Fh + 2Ph) / Sh) + 1

H2 = 252 - 3 + 0) / 1) + 1

H2 = 250

Spatial dimensions of the feature maps in the second layer = (250 x 250)

e

Number of weights for each filter = Fh x Fw x C1 x C2

Layer 1:
parameters: 5x5x1x32+32=832
image size:
W2 = ((W1 - Fw + 2Pw) / Sw) + 1
W2 = ((32 - 5 + 4)/1) + 1
W2 = 32

H2 = ((H1 - Fh + 2Ph) / Sh) + 1
H2 = ((32 - 5 + 4) / 1) + 1
H2 = 32

Layer 2:
parameters: 3x3x1x64+64=640
image size:
W2 = ((W1 - Fw + 2Pw) / Sw) + 1
W2 = ((32 - 3 + 2) / 1) + 1
W2 = 32

H2 = ((H1 - Fh + 2Ph) / Sh) + 1
H2 = ((32 - 3 + 2) / 1) + 1
H2 = 32

Layer 3:

Parameters:
3x3x64x128+128=73856

Image size:
W2 = ((W1 - Fw + 2Pw) / Sw) + 1
W2 = ((32 - 3 + 2) / 1) + 1
W2 = 32

H2 = ((H1 - Fh + 2Ph) / Sh) + 1
H2 = ((32 - 3 + 2) / 1) + 1

H1 = 32

Layer 4:
parameters:

HxWxC1xC2

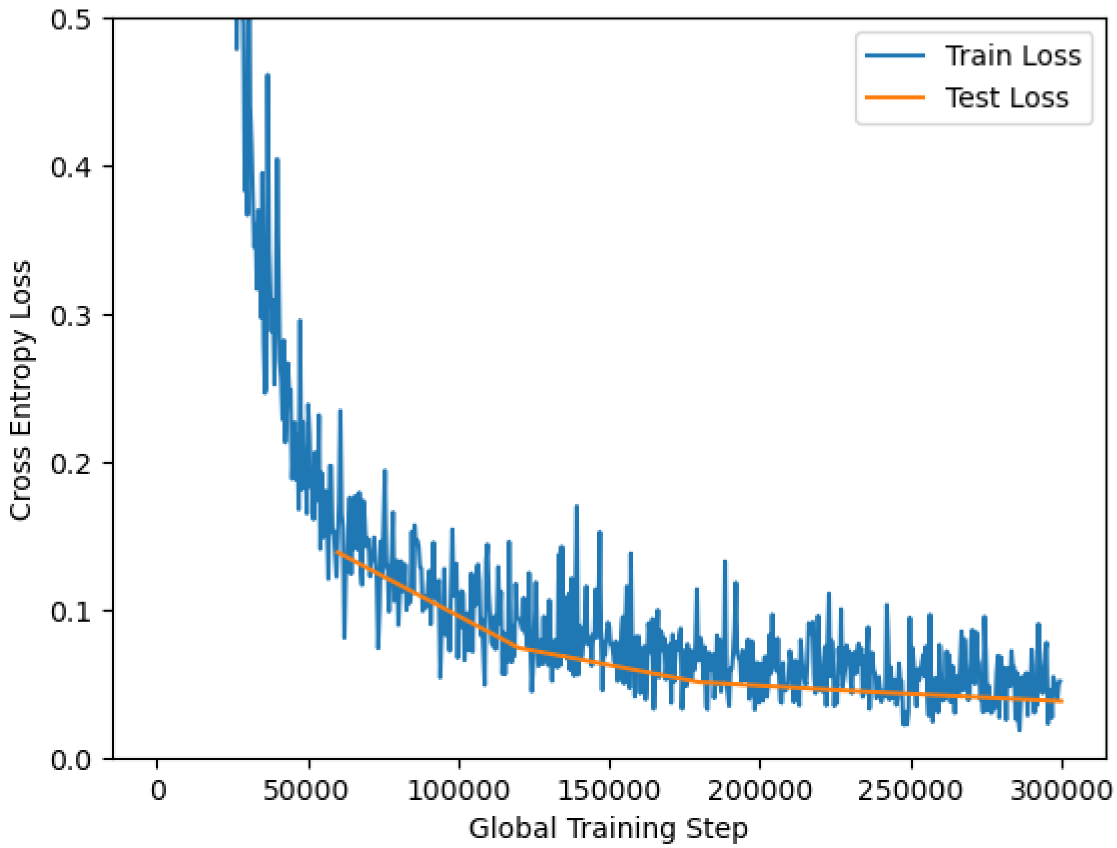
128x64+64=8256

Layer 5:
HxWxC1xC2
8256x1x10=82560

Total parameters:
832+640+73856+8256+82560=166144

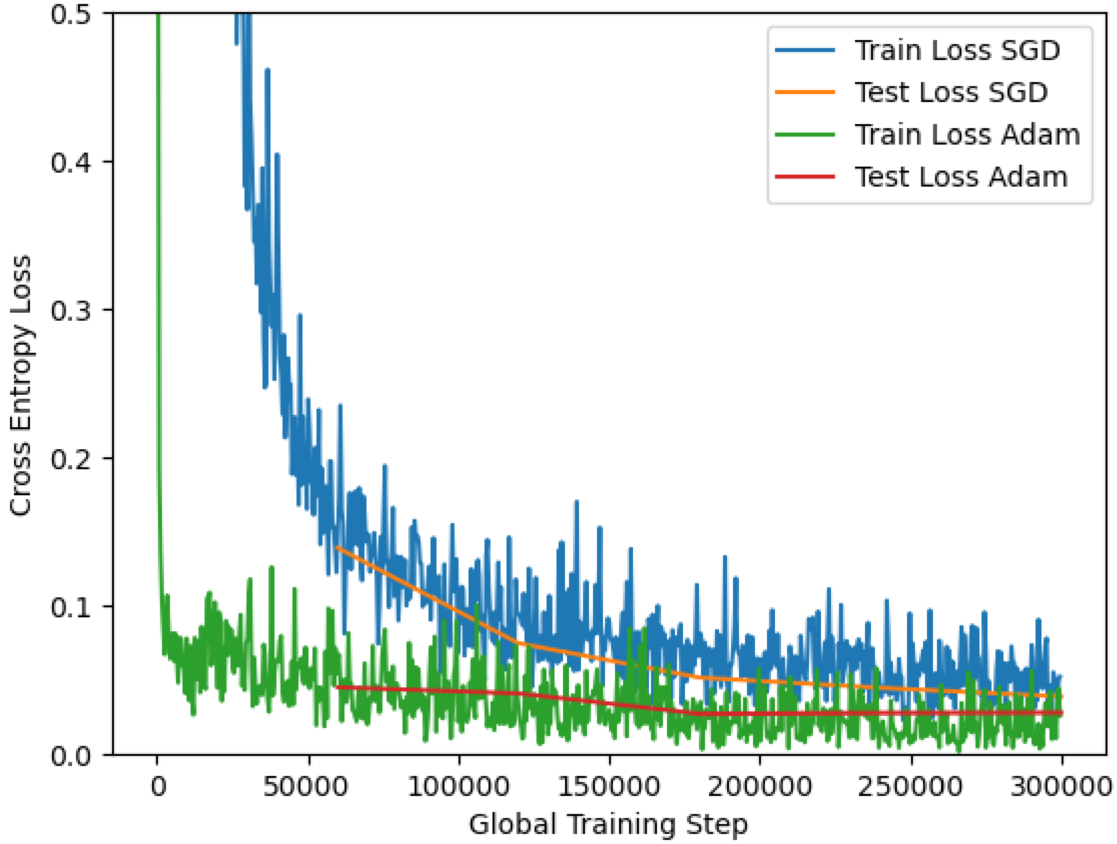
2

a

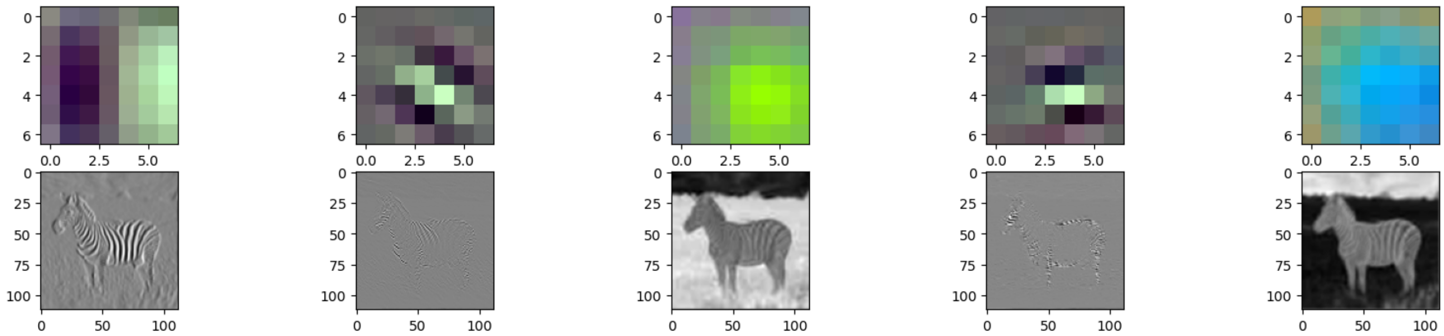


The model seems to **not** be overfitted since the loss during training only varies by less than 0.1 towards the final training steps.

b

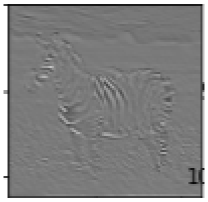


c

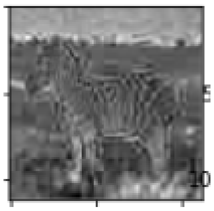


d

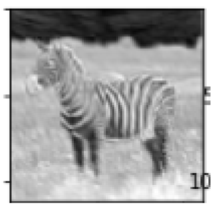
1. Diagonal line detection - mostly the diagonal lines of the zebra, as well as the horizon is visualized.



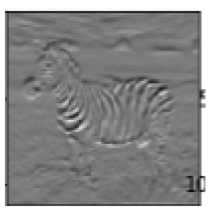
2. Low pass filter - image is smoothed out and loses a lot of detail.



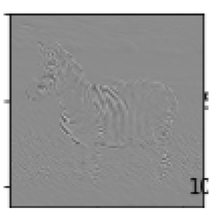
3. Inverted - the image has inverted colors, for example the white zebra lines are black and vice versa.



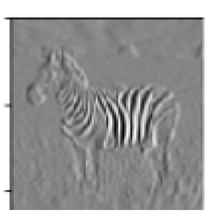
4. Horizontal line detection - the lines that are more horizontal are highlighted more than the vertically oriented ones (e.g. in the background).



5. High pass filter - the image retains only the details (lines) of the zebra.



6. Vertical line detection - the lines that are more vertical are highlighted more than the horizontally oriented ones. Less of the horizon in the background is visible here compared to in the horizontal line detection.



3

a

FFT works by looking at horizontal and vertical pixels separately and then combining them. The transformation itself works by comparing different sine/cosine frequencies with the frequencies of the brightness values in the spatial image.
In general, we can therefore already know that horizontal lines in the spatial domain will cause vertical lines in the frequency domain and vice versa. This is because horizontal lines will have no brighness variation in the horizontal direction, but will in the vertical direction, and vice versa.

In addition to this, the distance of the dots will be inversely proportional to the frequency of the sine/cosine frequency of the original image. This means that an input image with many lines (high frequency) will have a low distance between each point.

By this reasoning, we get these image pairs:

1a - 2e

1b - 2f

1c - 2c

1d - 2d

1e - 2b

1f - 2a

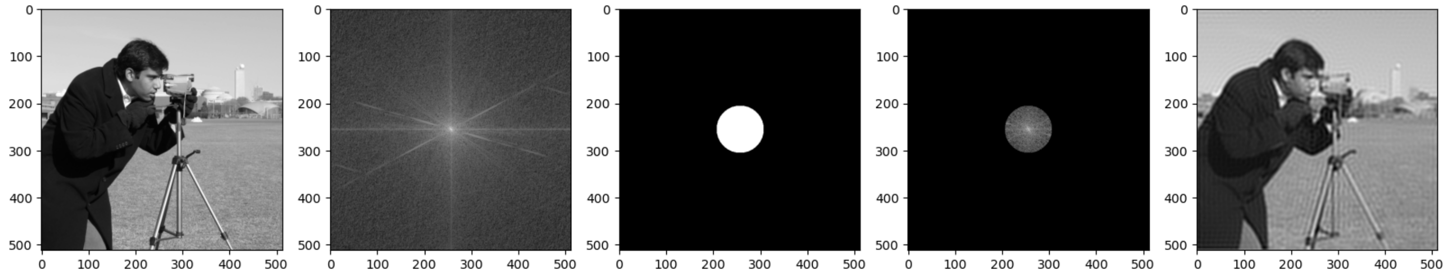
b

High pass and low pass filters are filters that generally only pass either high or low frequencies, respectively, through the filter. This means that for a high pass filter, only high frequencies will be visible, while a low pass filter will only show low frequencies. Lower frequencies in a Fast Fourier Transformed image are generally located in the middle, while higher frequencies are going more towards the corners. The kernels are represented as binary images with 1-s represented by white, and has a "passing" property, while the 0-s are represented by black and has a "blocking" property. This means that (b) is a low pass filter since it focuses on the middle, while (a) is a high pass filter since it focuses more on the opposite part of the image, especially the corners.

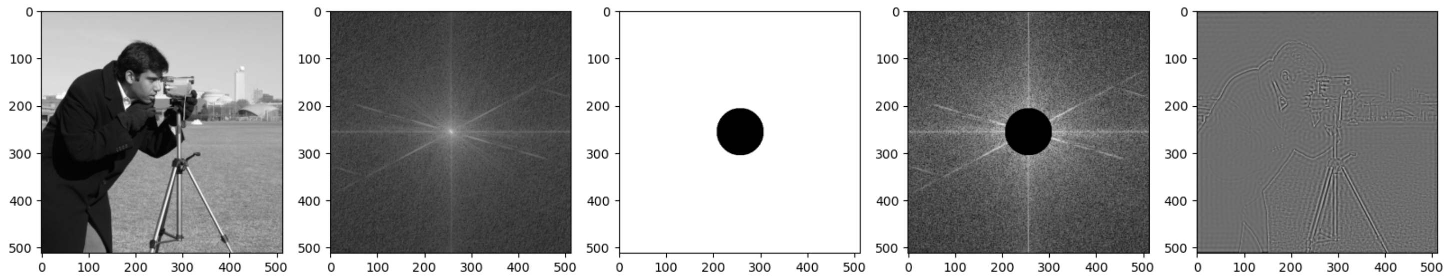
4

a

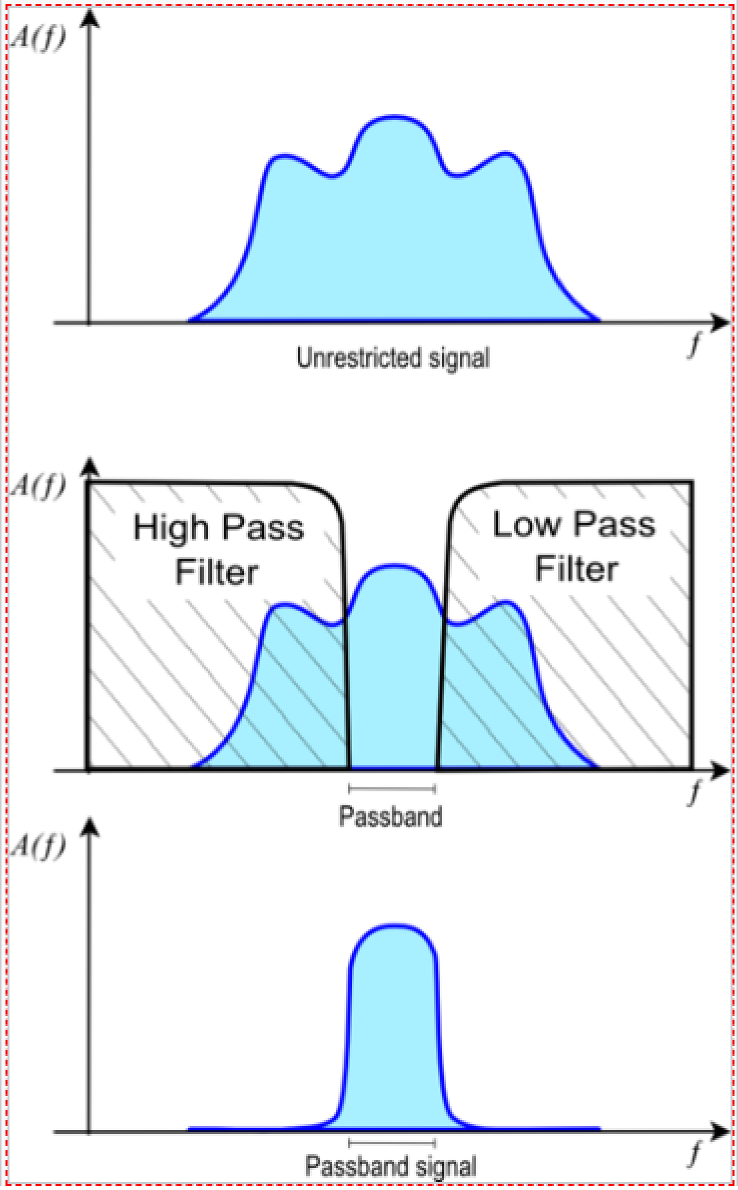
Low pass filtering:



High pass filtering:

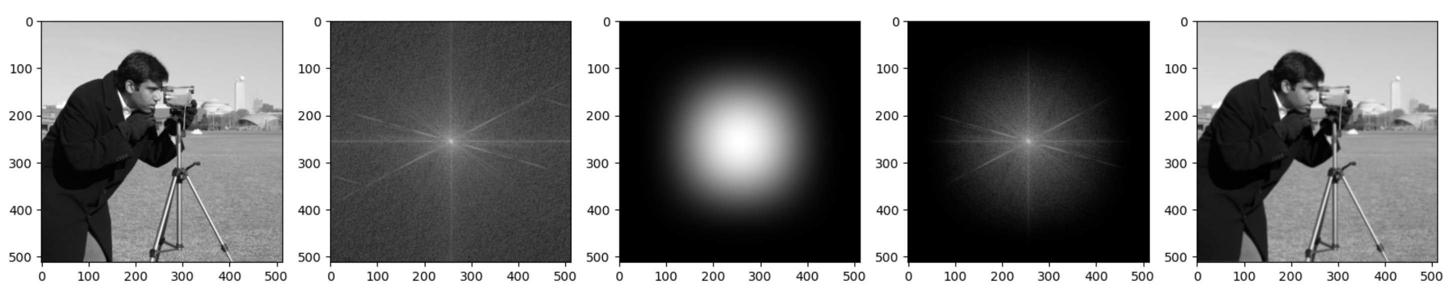


The ringing effect is an artifact that can appear in digital image processing. It visually looks like waves that emmit from sharp edges. It is called "ringing" because it is a wave-like oscillation similar to sound waves from a bell when it rings. It appears because the high pass and low pass filters have a sharp cutoff in frequencies that causes "instability" in the signal as it "rolls off". This can be seen here, were the resulting passband has very vertical lines, instead of a smooth roll off:

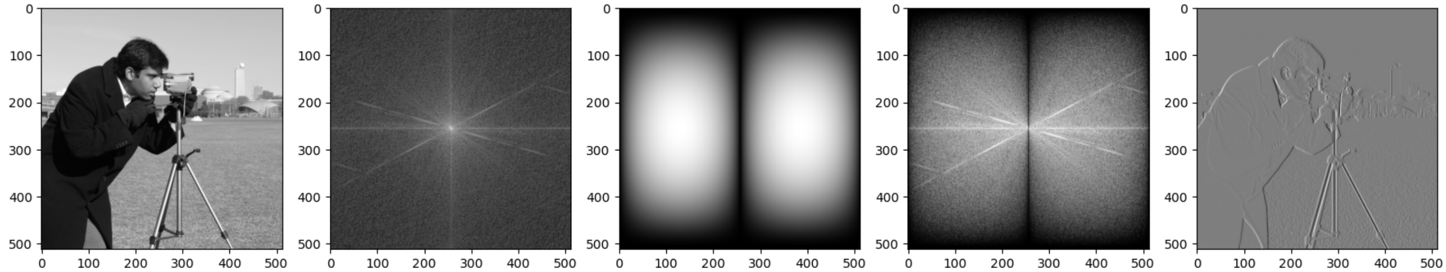


b

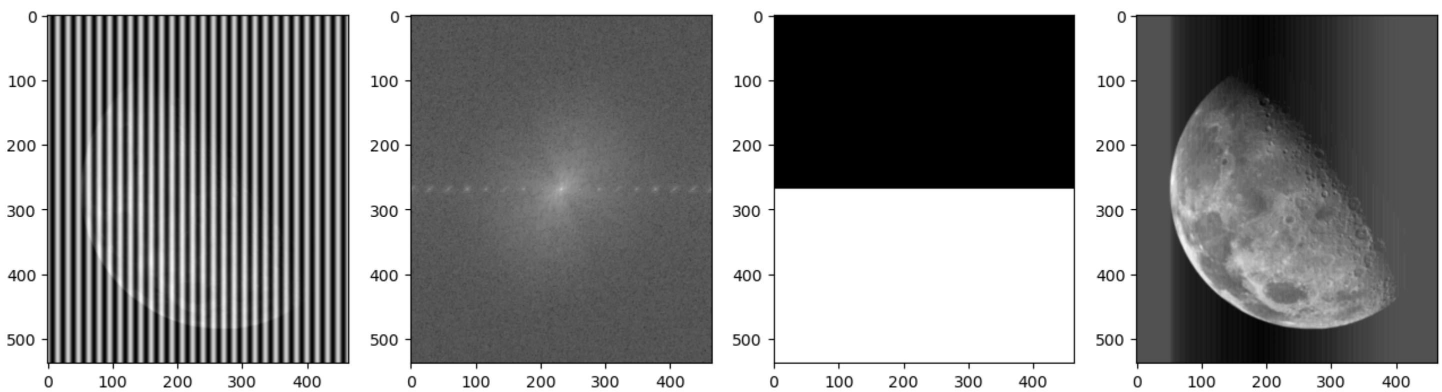
Gaussain filtering:



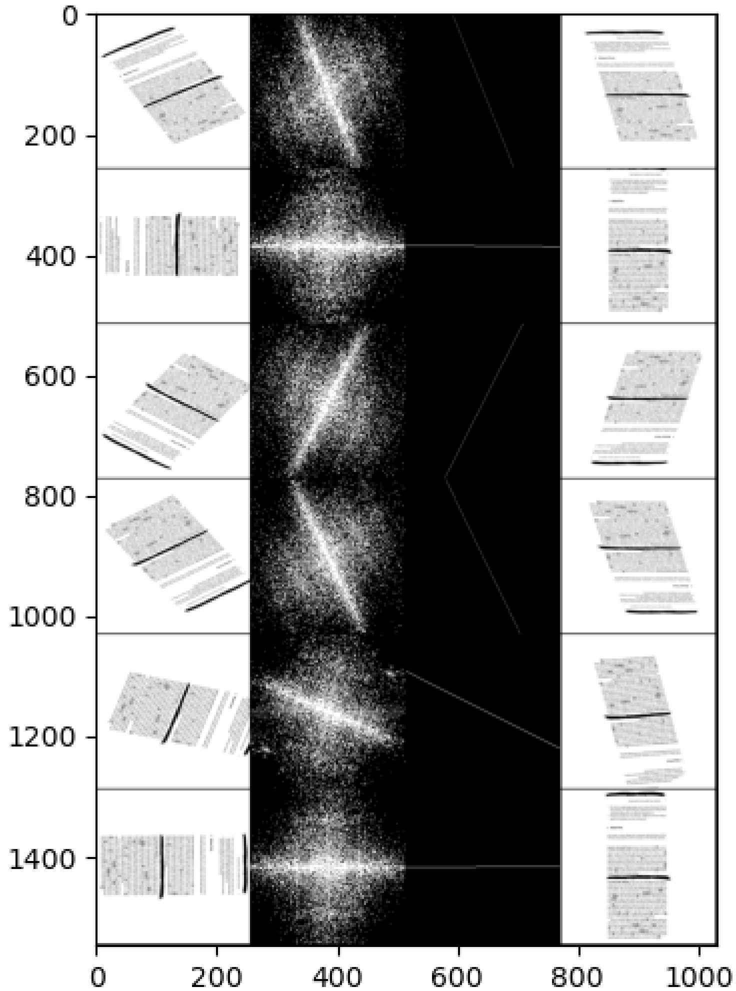
Sobel filtering:



c



d



0
200
400
600
800
1000
1200
1400
0
200
400
600
800
1000