Image Processing Exercise Book, TDT4195

Olav Olseng

October 14, 2013

Exercise 1

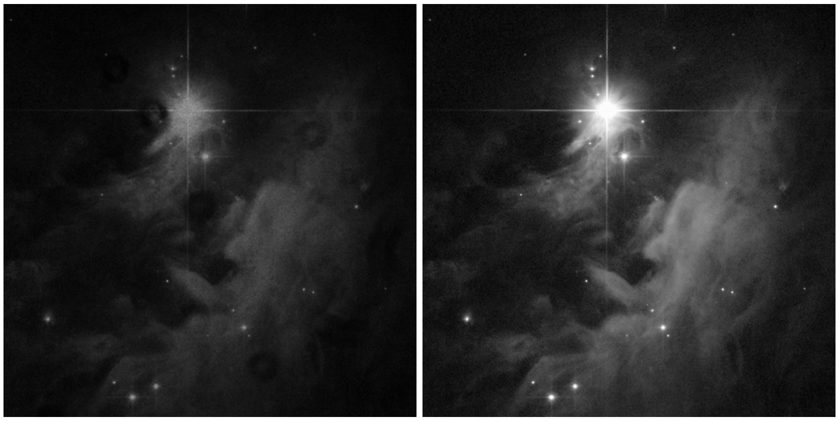
Basic Image Processing

Task 1

I first read the images converting to doubles by typing into the ”img” and ”flatField” variables. I then use the imdivide function from the images toolbox to get a pixel-by-pixel division, and I store the result in my ”res” variable. To show the result, i use the imshow function.

```
img = im2double(imread('testimages/disturbed_potw1144a.png'))
flatField = im2double(imread('testimages/flatfieldimage.png'))
res = imdivide(img, flatField)
imshow(res)
```

The images are loaded and normalized in a clamped double (values are between 0.0 and 1.0), with this loading function. The finalized images look like this:



(a) Original image

(b) Processed image

Task 2

We use division to *increase* the gray-level, if we subract it becomes darker. This is due to the fact that the images are normalized withing [0.0, 1.0]. So the blacker the value of the flatfield image is, the brighter the result image will be in that particular pixel. This can however, create problems if the flatfield image has pixel values of 0 (completely black), as we might get division by zero.

Filtering

Task 1

The image in the task is, at the time of writing, 17 pixels wide, not 16.
Original image:

0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0

(a) The result i get from applying the filter to the 1D image is this:

		0	0	1	2	3	4	5	4	3	2	1	0	0		

(b) There isn't any need to apply extra padding in this example, it already seems to be padded with the six 0s on each side of the five 1s in the original image. So there isn't any real loss of data. This really depends on the implementation of the filter however.

Task 2

If we do apply the continious convulution of this function we end up with:

(a)

$$\int_{-1/2}^{1/2} 1 \times 1 dv = 1 \tag{1}$$

(b) I expect this to give a the shape of a podium. Like they have in the olympics.

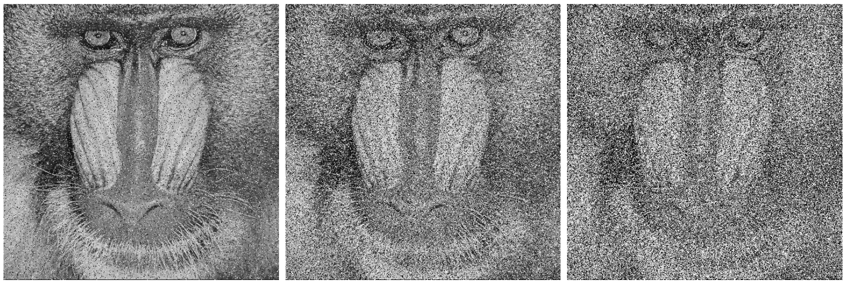
Noise

Task 1

I chose the mandrill picture for this assignment. I applied the different salt & pepper noise densities of 0.1, 0.3 and 0.5.
The noise was applied using the images toolbox function suggested in the assignment.

```
P = imnoise(I, 'salt & pepper', 0.1)
```

Figure 2: Different salt and pepper noise densities



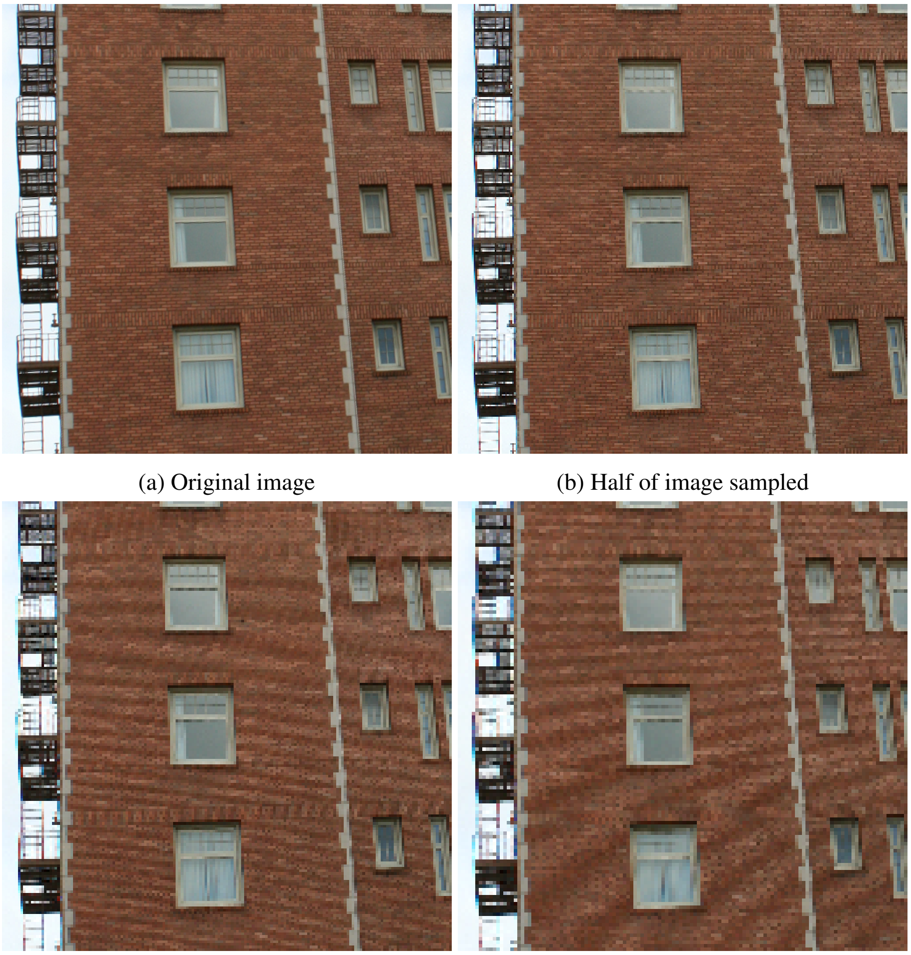
(a) 0.1 Density (b) 0.3 Density (c) 0.5 Density

Other noise types include, but not limited to, gaussian noise, uniform noise and exponential noise.

Aliasing

Task 1

Here are the brick images with different sampling rates:



(a) Original image

(b) Half of image sampled

(c) Third of image sampled

(d) Fourth of image sampled

As there is a pattern in the way the bricks a distributed in the wall, this pattern gets malformed as we downsample the image. Especially when we keep only a third and a fourth of the original data, we see new patterns emerge at the bottom of the wall. This effect was not as prevalent when i did the same downsampling on the lena.png image. This might be due to the fact that the image of a woman, doesn't have such a mathematical pattern in it, as the brickwall does.