1. (a) This proof will be shown for the $3 * 3$ kernel size but is the same for any $k > 1$. Consider the following separable 2D filter kernel g of size $2k + 1$ with $k = 1$.

$$\begin{aligned} g_1 &= \begin{bmatrix} w_4 & w_5 & w_6 \end{bmatrix} \\ g_2 &= \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \\ g &= g_2 g_1 \\ &= \begin{bmatrix} w_1 w_4 & w_1 w_5 & w_1 w_6 \\ w_2 w_4 & w_2 w_5 & w_2 w_6 \\ w_3 w_4 & w_3 w_5 & w_3 w_6 \end{bmatrix} \end{aligned}$$

Then, consider the following value for our "image" f, where the pixel of interest for this proof is x.

$$f = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & x & a_5 \\ a_6 & a_7 & a_8 \end{bmatrix}$$

Then, the convolution of f with g on the pixel x is:

$$\begin{aligned} h_{2D}(m, n) &= \sum_{k,l} g(k, l) f(m + k, n + l) \\ &= a_1 w_1 w_4 + a_2 w_1 w_5 + a_3 w_1 w_6 + a_4 w_2 w_4 + \\ & x w_2 w_5 + a_5 w_2 w_6 + a_6 w_3 w_4 + a_7 w_3 w_5 + a_8 w_3 w_6 \end{aligned}$$

Now, we must confirm that applying each of the 1D convolutions separately gives the same result. The convolution of f with g_1 is:

$$\begin{aligned} h_1 &= \sum_{l} g_1(l) f(m, n + l) \\ &= \begin{bmatrix} w_4 a_1 + w_5 a_2 + w_6 a_3 \\ w_4 a_4 + w_5 x + w_6 a_5 \\ w_4 a_6 + w_5 a_7 + w_6 a_8 \end{bmatrix} \end{aligned}$$

Then, the convolution of h_1 (the result from the previous step) with g_2 gives the following.

$$\begin{aligned} h_2 &= \sum_k g_2(k) h_1(m+k) \\ &= a_1 w_1 w_4 + a_2 w_1 w_5 + a_3 w_1 w_6 + a_4 w_2 w_4 + \\ & x w_2 w_5 + a_5 w_2 w_6 + a_6 w_3 w_4 + a_7 w_3 w_5 + a_8 w_3 w_6 \end{aligned}$$

Thus, the result is the same in both cases, so convolving an image with a discrete, separable 2D filter kernel is equivalent to convolving with two 1D filter kernels.

(b) For each pixel, a 2D filter kernel would perform $(2k+1)^2$ operations, while each 1D filter kernel would perform $2k+1$ operations. So, the number of operations saved per pixel is:

$$\begin{aligned} (2k+1)^2 - 2 * (2k+1) &= (4k^2 + 4k + 1) - (4k+2) \\ &= 4k^2 - 1 \end{aligned}$$

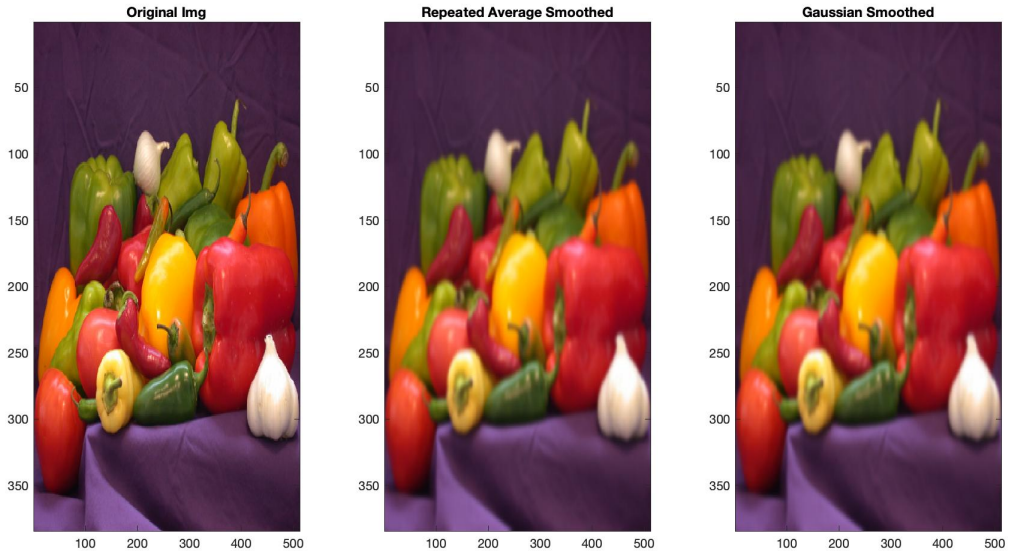
Based on this, the total number of operations saved for an $N * N$ image is $\boxed{N^2(4k^2-1)}$.

2.

3.

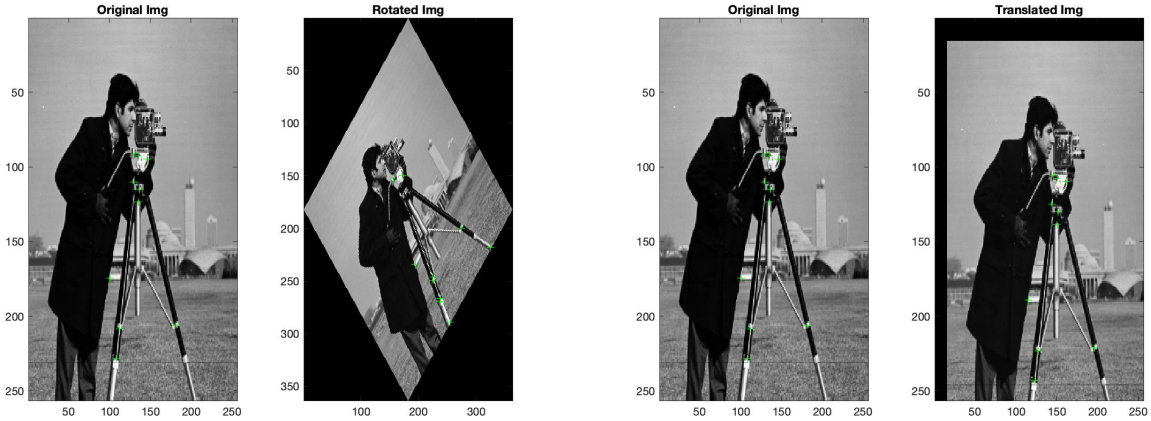
4.

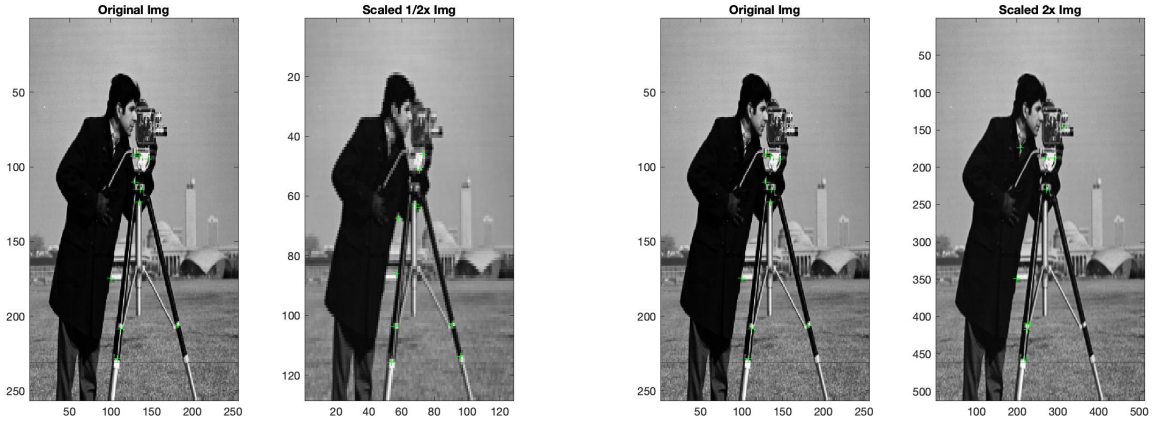
5. A way to verify that repeatedly applying an averaging filter will approximate Gaussian smoothing is to take an image and apply both side by side. First, apply a Gaussian smooth to it, then take the original image and apply a averaging filter with a small kernel size like 3x3. Then repeat the application of the averaging filter until the output image looks approximately like the result of the Gaussian smoothing. Then, if we take the root mean square error of the pixels for both images compared to the original image and take the average of their difference, we can have an average RMS error for the difference between Gaussian smoothing and repeated averaging. As you can see from the plot below, the error is very small meaning the Gaussian smoothing with sigma=2 and using a 3x3 averaging filter 5 times are very close to the same effect.



Average Difference in RMS Error of all pixels between Gaussian and Average: 0.051153

6. (a) text

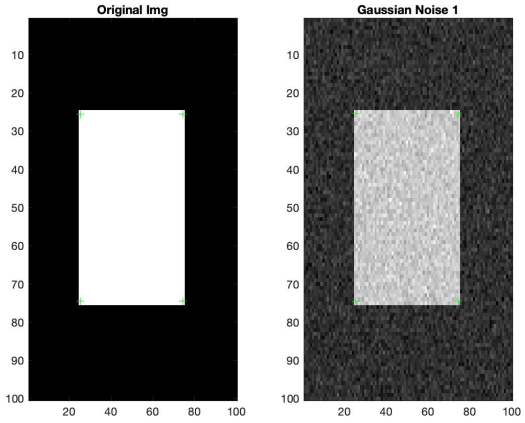


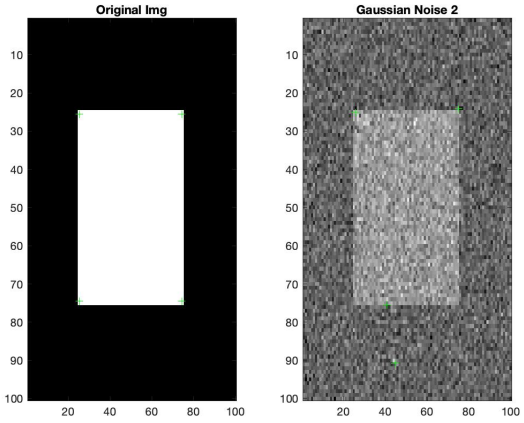

Original Img
Scaled 1/2x Img
Original Img
Scaled 2x Img

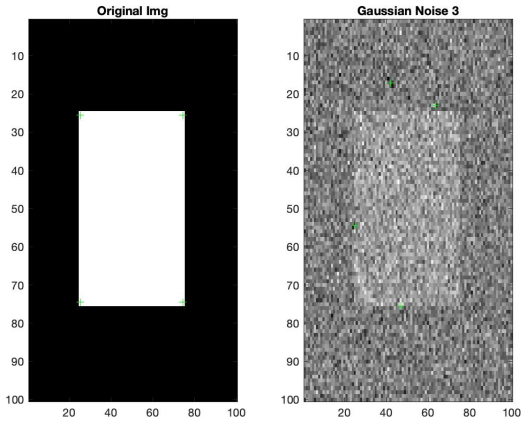
(b) text2

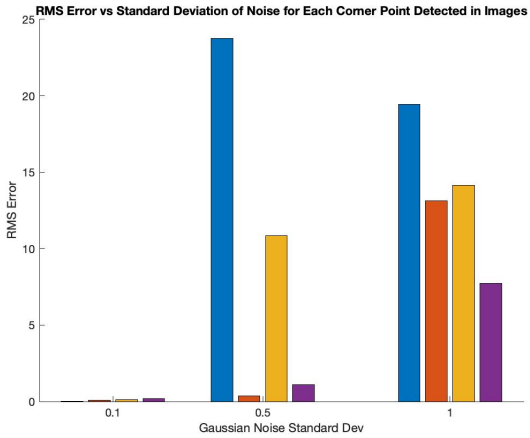

Original Img
Brightened Img
Original Img
Darkened Img
Original Img
Sharpened 3x3 Img
Original Img
Sharpened 5x5 Img

(c) text3


Original Img
Gaussian Noise 1


Original Img
Gaussian Noise 2


Original Img
Gaussian Noise 3


RMS Error vs Standard Deviation of Noise for Each Corner Point Detected in Images
RMS Error
Gaussian Noise Standard Dev
0.1
0.5
1

Matlab Code