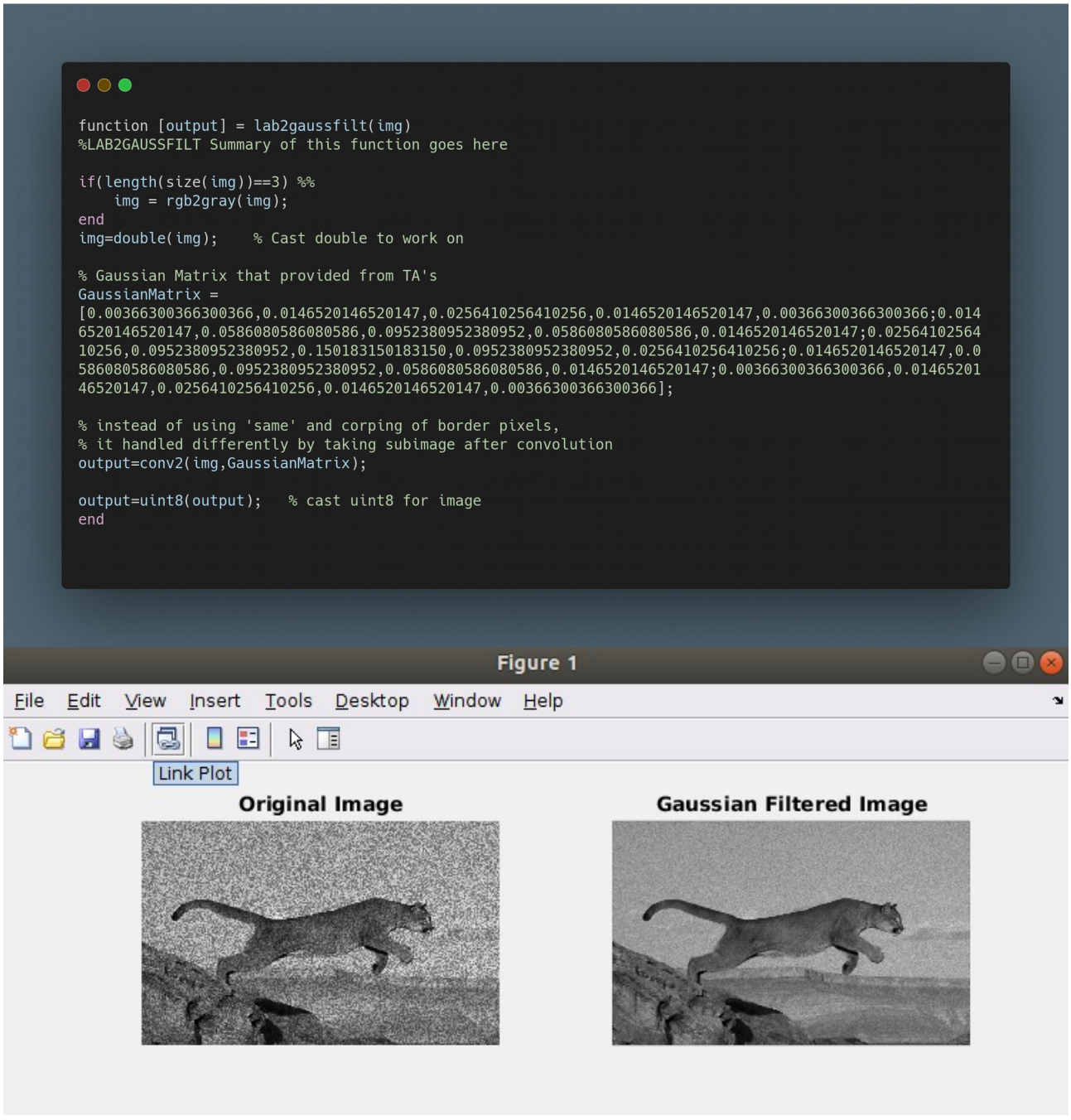
EE417- Computer Vision Lab 2 Report
Hakan Buğra Erentuğ 23637
12 / 08 / 19

Assignment 1:

In this assignment students are expected to implement a linear filter called Gauss Filter which is made for smoothing the image and reduces the noises. In the task 1, the kernel size was already defined to 5 x 5. It is possible to enlarge the kernel to smooth image more, however it also reduces the the sharpness and clearness of the image. The proper .mat file to carry the information of the carnel has already given to students too. Our task is importing this kernel and utilize it to create Gaussian smoothed version of '*jump.png*' (Check: Appendix 1). This time, students allowed to use 'conv2()' built-in defined MATLAB function.

My implementation of Gaussian filter has given below.

```
function [output] = lab2gaussfilt(img)
%LAB2GAUSSFILT Summary of this function goes here

if(length(size(img))==3) %%
    img = rgb2gray(img);
end
img=double(img);    % Cast double to work on

% Gaussian Matrix that provided from TA's
GaussianMatrix =
[0.00366300366300366,0.0146520146520147,0.0256410256410256,0.0146520146520147,0.00366300366300366;0.014
6520146520147,0.0586080586080586,0.0952380952380952,0.0586080586080586,0.0146520146520147;0.02564102564
10256,0.0952380952380952,0.150183150183150,0.0952380952380952,0.0256410256410256;0.0146520146520147,0.0
586080586080586,0.0952380952380952,0.0586080586080586,0.0146520146520147;0.00366300366300366,0.01465201
46520147,0.0256410256410256,0.0146520146520147,0.00366300366300366];

% instead of using 'same' and corping of border pixels,
% it handled differently by taking subimage after convolution
output=conv2(img,GaussianMatrix);

output=uint8(output);   % cast uint8 for image
end
```



EE417- Computer Vision Lab 2 Report
Hakan Buğra Erentuğ 23637
12 / 08 / 19

Self-Criticism: The one thing that I can do differently is handling the border pixels in the function. I was allowed to use conv2(A,B) function, however maybe I can add one more parameter and use conv2(A,B,'same');. Instead, after I call the Gaussian function that I defined, I take the subimage of main image in order to corp the borders. Due to defined size of the kernel, it is just improvement, not a correction.

Assignment-2:

In the second task, we are expected to implement a non-linear filter that called median filter. The median filtering is used of the eliminate salt&pepper noises. We also implemented the Gaussian filtering to see the difference. However, In theory and practice, Gaussian filter cannot eliminate the salt&pepper noise. The irrelevant low and high valued pixels, just will be imbued the neighbor pixels and make the image blurred. The implementation of median filter and the test results of the image '*tiger.png*' are presented below (Check: Appendix 2).

```
function ret = lab2medfilt(img, k)
% k stands for square kernel size
if(length(size(img))==3)
    img = rgb2gray(img);    % GRAY-SCALE
end

[r, c] =size(img);
new=zeros(r,c);

img = double(img);

for i =k+1:1:r-k-1
    for j=k+1:1:c-k-1

        subimg=img(i-k:i+k, j-k:j+k); % corp image manually.
        value = median(subimg(:));  % Take median of filter to assign to pixel
        new(i,j)=value;

    end
end

ret = uint8(new);
end
```

EE417- Computer Vision Lab 2 Report
Hakan Buğra Erentuğ 23637
12 / 08 / 19

Assignment 3:

In the third task, students are trying to sharpen the image. To sharpen the image, we have to generate smoothed image and subtract the pixels values with proper coefficients which is called lambda. The generalized formula is $J(p) = I(p) + \lambda [I(p) – S(p)]$. In the function, we take one more input that called M to decide the which smoothing method will be used from the set of Gaussian, median and mean filtering. The Gaussian and median filter already implemented in assignment 1 and assignment 2, however the mean box filtering have taken from previous lab folders. The high λ *values sharpen the image more.* Implementation of sharpening and mean filter with test results are placed below. The used test image is '*mother.png*' (Check: Appendix-3).

Local Mean Box Filter:

```
function ret = lab2locbox(img, k)

if(length(size(img))==3)
    img = rgb2gray(img);    %%Convert to gray scale
end
[r, c] =size(img);
new=zeros(r,c);
img = double(img);  %% cast double to work on it

for i =k+1:1:r-k-1
    for j=k+1:1:c-k-1

        subimg=img(i-k:i+k, j-k:j+k);
        value = mean(subimg(:));    % take mean of kernel and assign
        new(i,j)=value;

    end
end

ret = uint8(new);      %% convert to 8bit integer to show image

end
```

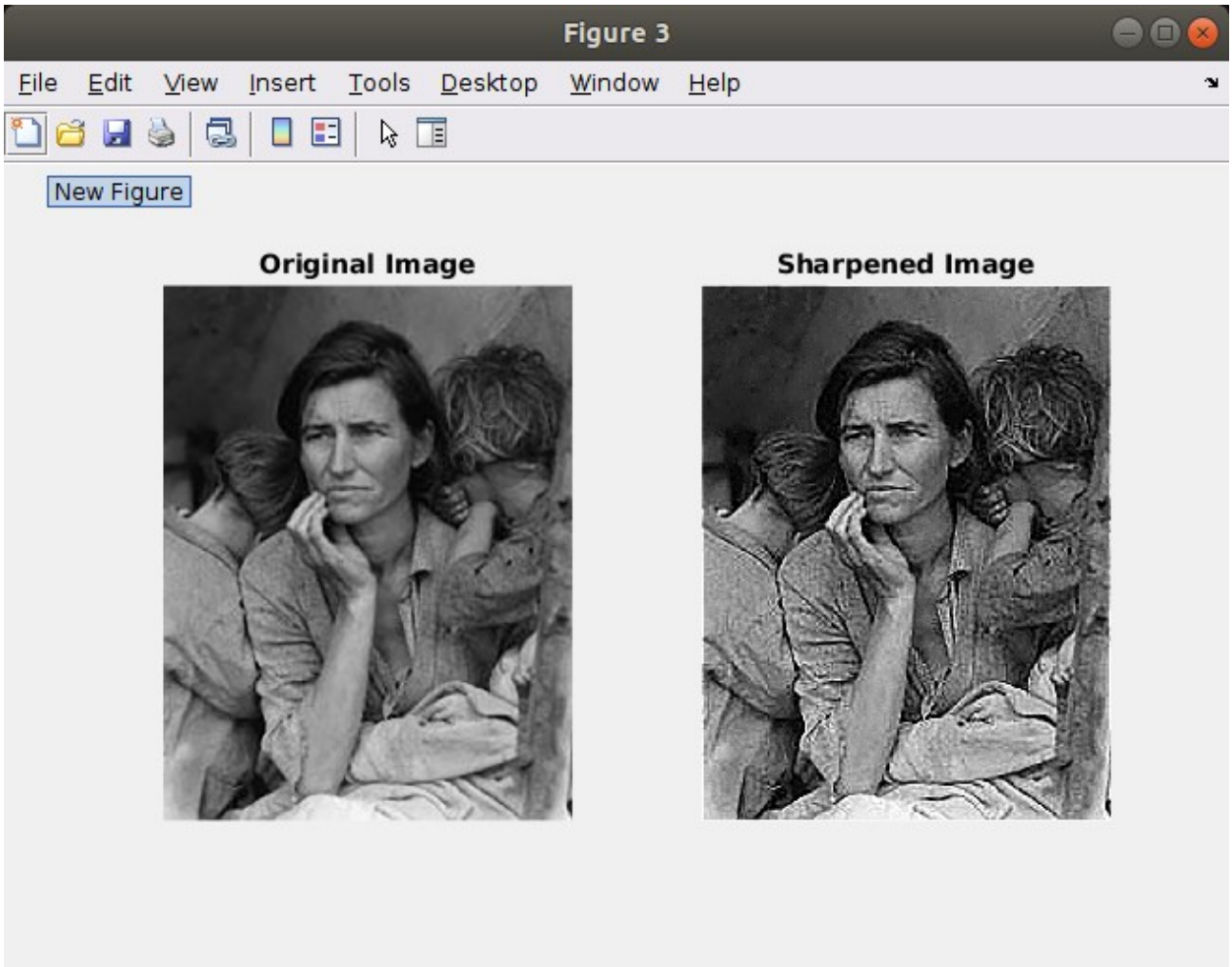
EE417- Computer Vision Lab 2 Report
Hakan Buğra Erentuğ 23637
12 / 08 / 19

The Sharpening Function:

```
function [ret] = lab2sharpen(img,lambda,M)
%LAB2SHARPEN Summary of this function goes here
%   Detailed explanation goes here
% M ==>  1 for local box, 2 for Gaussian, 3 for median
if(length(size(img))==3)
    img = rgb2gray(img);
end

img = double(img);

if(M==1)
    soft=lab2locbox(img,3);     % kernel size assumed 3
end

if(M==2)
    soft=lab2gaussfilt(img);
    [r,c]=size(soft);
    soft=soft(3:r-2, 3:c-2);    % Subimage has taken to delete borders which
                                % comes from gaussian filter convolutions.
end

if(M==3)
    soft=lab2medfilt(img,3);    % kernel size assumed 3
end

soft=double(soft);  % soft stands for smoothened image


ret= img + (lambda.*(img-soft)); % To sharpen the image

ret=uint8(ret);             % uint8 convertion

end
```

EE417- Computer Vision Lab 2 Report
Hakan Buğra Erentuğ 23637
12 / 08 / 19

The Test Results with Lambda value 10 and M value 2 which means Gaussian filter:



EE417- Computer Vision Lab 2 Report
Hakan Buğra Erentuğ 23637
12 / 08 / 19

Assignment 4:

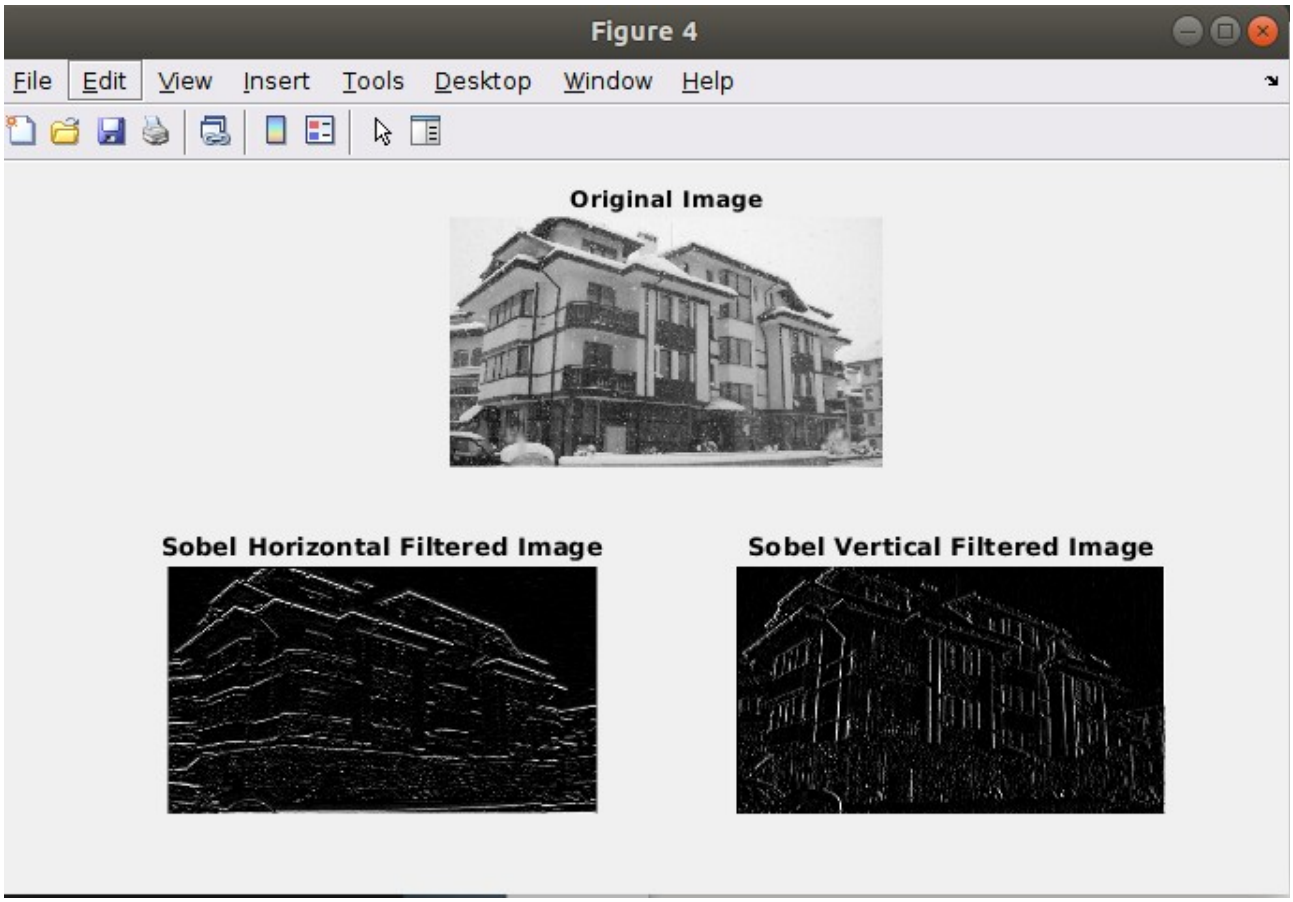
In the last task, we are trying to use the sobel algorithm to make clarified the edges of image. We We use the kind of discrete derivative filter to detect the rapid changes in pixel values. The crucial part is smoothing the image before filtering. The following kernels could be applied:

X Filter				Y Filter		
-1	0	1		-1	-2	-1
-2	0	2		0	0	0
-1	0	1		1	2	1

The followings are the test result of image '*house.png*' (Check: Appendix-4) and implementation of function.

```
function [ret] = lab2sobelfilt(img, XorY)
%LAB2SOBELFILT Summary of this function goes here

% XorY stands for horitontal or vertical sober operation
% 'X' for X filter and, 'Y' for Y filter

if(length(size(img))==3)
    img = rgb2gray(img);
end
img=double(img);

img=lab2gaussfilt(img);

if(XorY=='X')
    filter= [-1 0 1; -2 0 2; -1 0 1];
end
if(XorY=='Y')
    filter= [-1 -2 -1; 0 0 0; 1 2 1];
end

ret=conv2(img,filter);
ret=uint8(ret);
[r,c]=size(ret);
ret=ret(3:r-2, 3:c-2);    % To eliminate border which comes
                          % from gaussian filter

end
```

EE417- Computer Vision Lab 2 Report
Hakan Buğra Erentuğ 23637
12 / 08 / 19



You can find the main.m in the Appendix-5, caller and executer function of all other functions.

Self-Criticism: In spite of previous lab report, I corrected the many mistake that I did such as comments, casting or RGB to gray-scale. The only unsatisfactory design of function that I noticed is the fact that my Gaussian Filter implementation changes the size of the image due to convolution without ‘same’ parameter. Therefore, I have to take the sub image to re-change the image to normal size. I was able to do that and see my mistake as an insufficient way rather than incorrect way thanks to fixed Gaussian kernel size that provided from TA’s. If the forming the kernel would become our job, I could not do that. Thank you for sparing time.

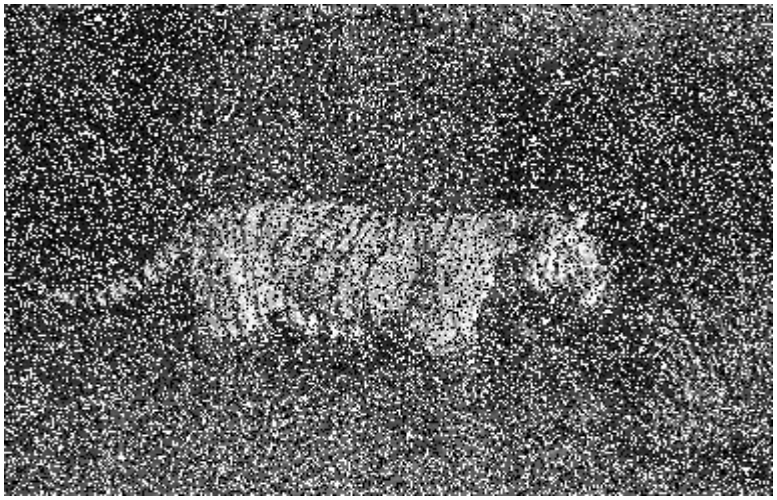
EE417- Computer Vision Lab 2 Report
Hakan Buğra Erentuğ 23637
12 / 08 / 19

Appendices:

Appendix-1

jump.png

Appendix-2



Tiger.png

EE417- Computer Vision Lab 2 Report
Hakan Buğra Erentuğ 23637
12 / 08 / 19

Appendix 3

mother.png

EE417- Computer Vision Lab 2 Report
Hakan Buğra Erentuğ 23637
12 / 08 / 19

Appendix 4

house.png

EE417- Computer Vision Lab 2 Report
Hakan Buğra Erentuğ 23637
12 / 08 / 19

Appendix 5

```
clear;
clc;
close all;

% Hakan Buğra Erentuğ Lab-2 the main caller function 09-10-19
% Task 1

img = imread('jump.png');
figure;subplot(1,2,1);imshow(img);title('Original Image');
subplot(1,2,2);
out=lab2gaussfilt(img);
imshow(out);title('Gaussian Filtered Image');

% Task 2

img2 = imread('Tiger.png');
figure;subplot(1,3,1);imshow(img2);title('Original Image');
subplot(1,3,2);
out=lab2gaussfilt(img2);
imshow(out);title('Gaussian Filtered Image');
subplot(1,3,3);
out=lab2medfilt(img2,3);    % kernel size 3
imshow(out);title('Median Filtered Image');

% Task 3

img3 = imread('mother.png');
figure;
subplot(1,2,1);imshow(img3);title('Original Image');
subplot(1,2,2);
out=lab2sharpen(img3,10,2); % lambda= 10, 2 for gauss filter
imshow(out);title('Sharpened Image');

% Task 4
img4 = imread('house.png');
 figure;
 x(1)=subplot(2,2,1);imshow(img4);title('Original Image');
 x(2)=subplot(2,2,2);
 posn=mean(cell2mat(get(x(1:2),'position')));   % Merge first two frame and get average location
 delete(x(2)); x(2)=[];                         % delete the second frame.
 set(x(1),'position',posn)                      % Set first to average location

 subplot(2,2,3);imshow(lab2sobelfilt(img4,'Y'));title('Sobel Horizontal Filtered Image');
 subplot(2,2,4);imshow(lab2sobelfilt(img4,'X'));title('Sobel Vertical Filtered Image');
```

main.m