Assignment 2: Image Filters

Elijah Garmon,

Florida Polytechnic University

Department of Computer Science
Florida Polytechnic University, Florida, USA

February 2024

ABSTRACT

This assignment is all about different image filters. Many of these filters will created manually and through the built-in functions inside OpenCV. All of these filters will also be applied using both a 3x3 and a 5x5 image window size.

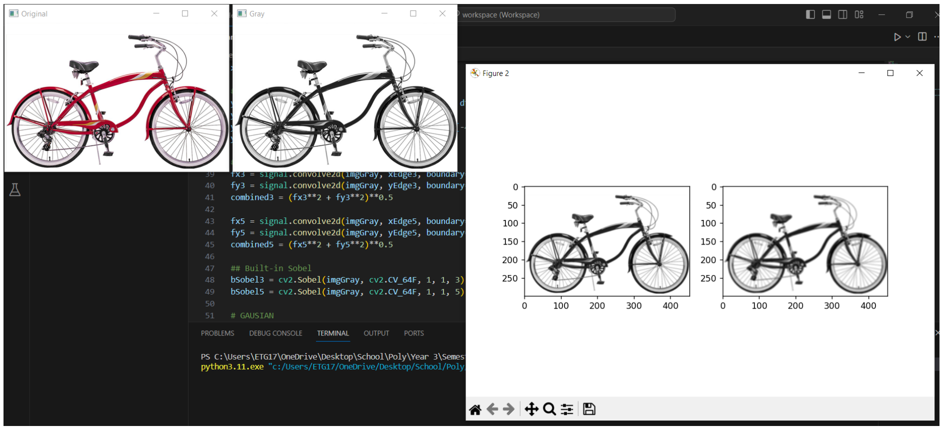
CONTENTS

LIST OF FIGURES

1 BOX FILTER

1.1 BOX FILTER (MANUAL)

A box filter is used to help with image smoothing. It removes any outliers the image might have. A box filter does a mean operator on a 3x3 window or a 5x5 to perform this action. To do this manually, I used a 3x3 mask and a 5x5 mask, both of which are set to a division of 255, which is applied using our main image using the convolve2d using SciPy library. For these images to appear, I used graphed the image using the matplotlib library.



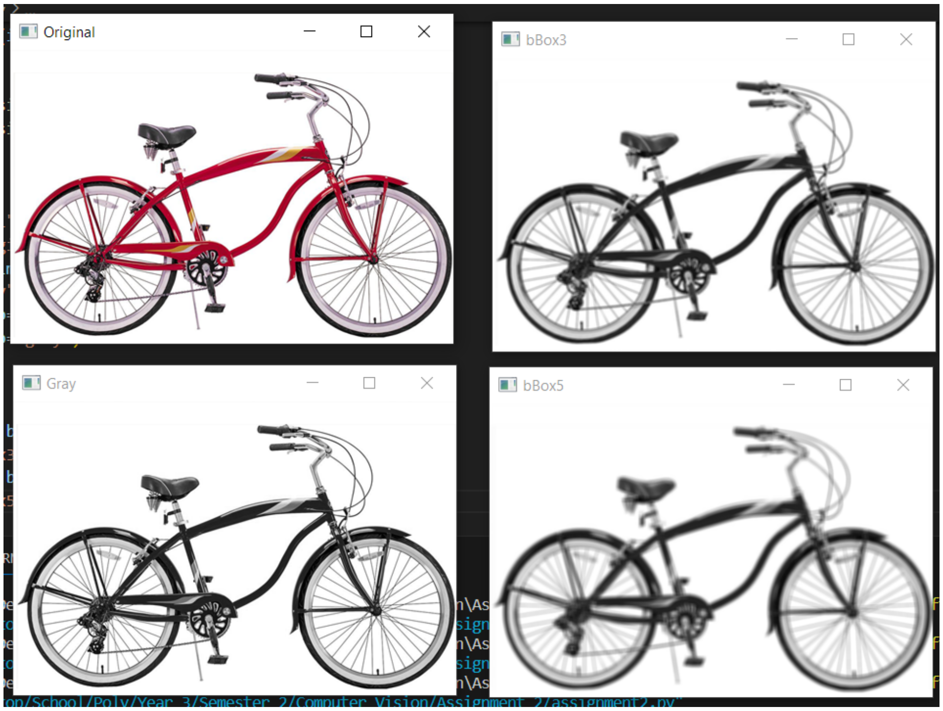
```
fx3 = signal.convolve2d(imgGray, xEdge3, boundary
fy3 = signal.convolve2d(imgGray, yEdge3, boundary
combined3 = (fx3**2 + fy3**2)**0.5

fx5 = signal.convolve2d(imgGray, xEdge5, boundary
fy5 = signal.convolve2d(imgGray, yEdge5, boundary
combined5 = (fx5**2 + fy5**2)**0.5

## Built-in Sobel
bSobel3 = cv2.Sobel(imgGray, cv2.CV_64F, 1, 1, 3)
bSobel5 = cv2.Sobel(imgGray, cv2.CV_64F, 1, 1, 5)

# GAUSIAN
```

```
PS C:\Users\ETG17\OneDrive\Desktop\School\Poly\Year 3\Semes
python3.11.exe "c:/Users/ETG17/OneDrive/Desktop/School/Poly
```

Figure 1.1. Manual Box Filters. (Ordered left to right) Original Image, Gray Image, 3x3 Box Filter, 5x5 Box Filter

1.2 BOX FILTER (OPEN CV)

To use the built-in box filter from OpenCV, all that was required was to run the boxFilter command. This command is run as a (3,3) and a (5,5). Since this was done using only OpenCV, I did not need to graph it onto a matplotlib graph and instead could output the image into a window as if it were a normal image.



```
on/School/Poly/Year 3/Semester 2/Computer Vision/Assignment 2/assignment2.py"
```

Figure 1.2. Built-in Box Filters. Original Image (top left), 3x3 Box Filter (top right), Gray Image (bottom left), 5x5 Box Filter (bottom right

2 SOBEL FILTER

2.1 SOBEL FILTER (X-AXIS)

A Sobel filter is used for edge detection, and there are two main methods: using the x-axis and the y-axis. A Sobel filter requires multiplying the original image by an array.

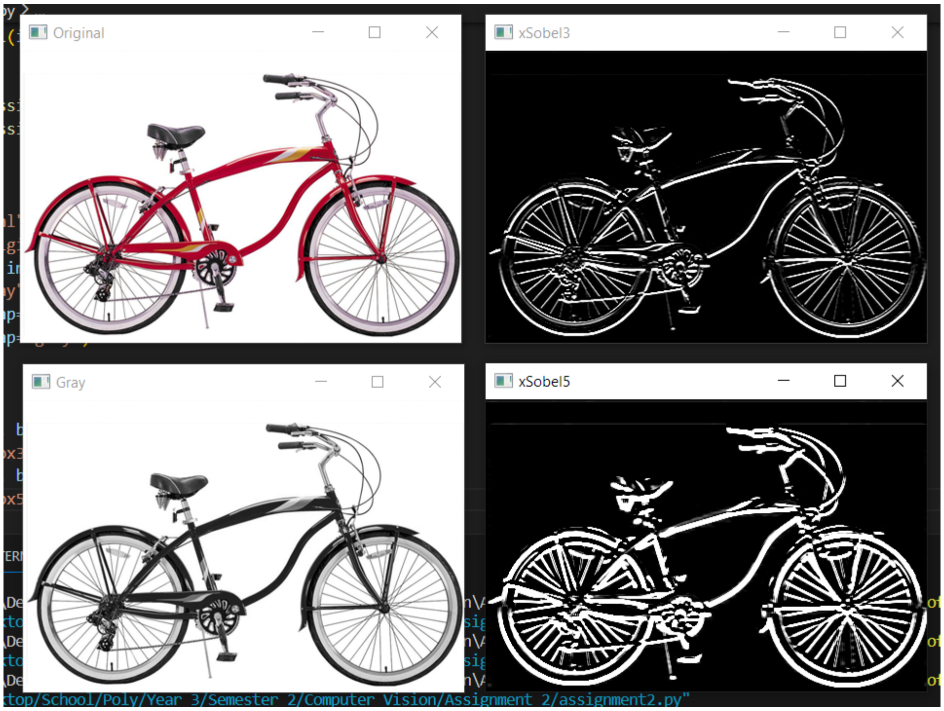
For a 3x3 x-axis, I used the array:

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

And for a 5x5 x-axis, I used the array:

$$\begin{bmatrix} -2 & -2 & -4 & -2 & -2 \\ -1 & -1 & -2 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 2 & 1 & 1 \\ 2 & 2 & 4 & 2 & 2 \end{bmatrix}$$

To merge the arrays and the image, I used the filter2D function.



```
top/School/Poly/Year 3/Semester 2/Computer Vision/Assignment 2/assignment2.py"
```

Figure 2.1. X-axis Sobel Filters. Original Image (top left), 3x3 X-axis Sobel Filter (top right), Gray Image (bottom left), 5x5 X-axis Sobel Filter (bottom right)

2.2 SOBEL FILTER (Y-AXIS)

Like the X-Axis Sobel filter, I used the same filter2D function to combine the different arrays with the images.

For a 3x3 y-axis, I used the array:

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

And for a 5x5 y-axis, I used the array:

$$\begin{bmatrix} -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -4 & -2 & 0 & 2 & 4 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \end{bmatrix}$$

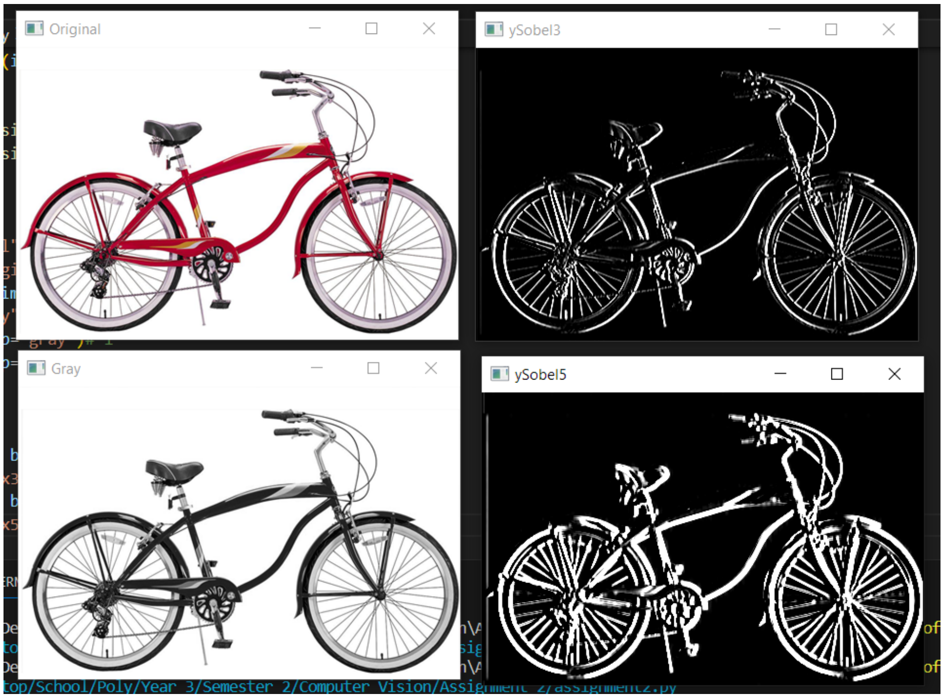

Original
ySobel3
Gray
ySobel5
top/School/Poly/Year 3/Semester 2/Computer Vision/Assignment 2/assignment2.py

Figure 2.2. Y-axis Sobel Filters. Original Image (top left), 3x3 Y-axis Sobel Filter (top right), Gray Image (bottom left), 5x5 Y-axis Sobel Filter (bottom right)

2.3 SOBEL FILTER (X-AXIS AND Y-AXIS)

To merge the X-Axis and Y-Axis Sobel filters, I first found the derivative of the images using the convolve2d using the SciPy library. I then did the square of the X-axis derivative and the square of the Y-axis derivative, added them together, and then found the square root.

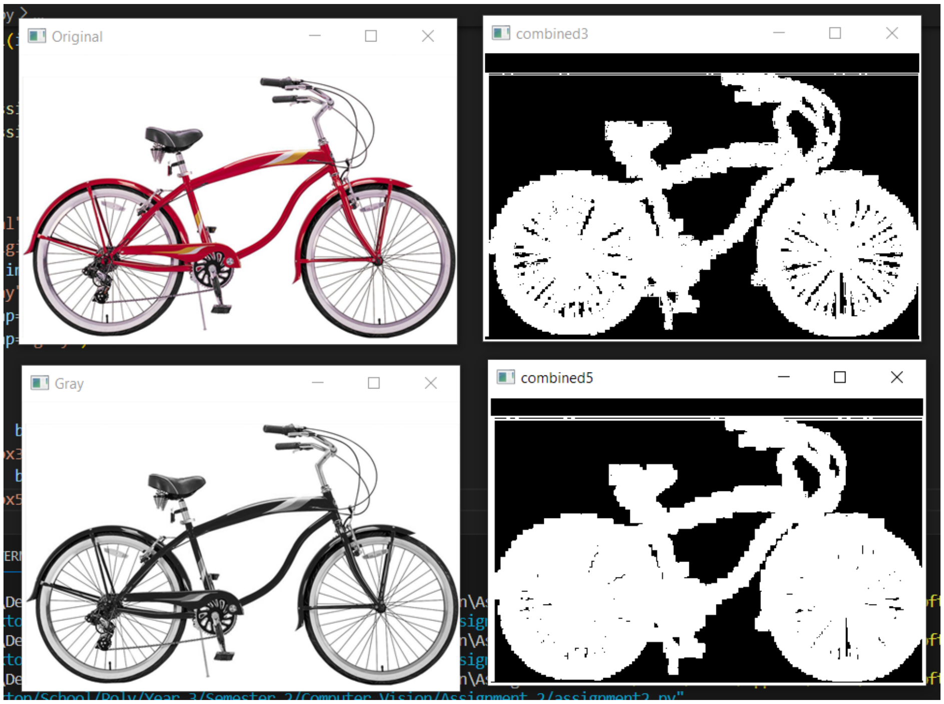
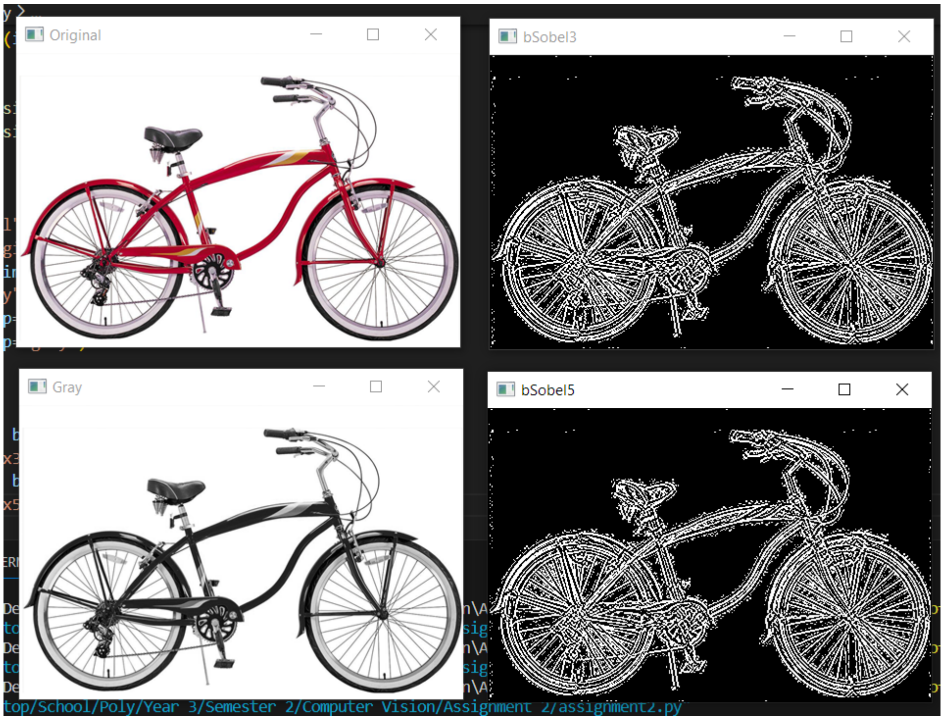


Figure 2.3. Combined Sobel Filters. Original Image (top left), 3x3 Sobel Filter (top right), Gray Image (bottom left), 5x5 Sobel Filter (bottom right)

2.4 SOBEL FILTER (OPENCV)

To do the built-in OpenCV sobel filter, I used the sobel function. I applied it in the x and the y for k=3 and k=5.



```
top/School/Poly/Year 3/Semester 2/Computer Vision/Assignment 2/assignment2.py
```

Figure 2.4. Built-in Sobel Filters. Original Image (top left), 3x3 Sobel Filter (top right), Gray Image (bottom left), 5x5 Sobel Filter (bottom right)

3 GAUSSIAN FILTER

We only needed to apply the filter for the Gaussian Filter using the built-in OpenCV function. I applied the function using a 3x3 and a 5x5.

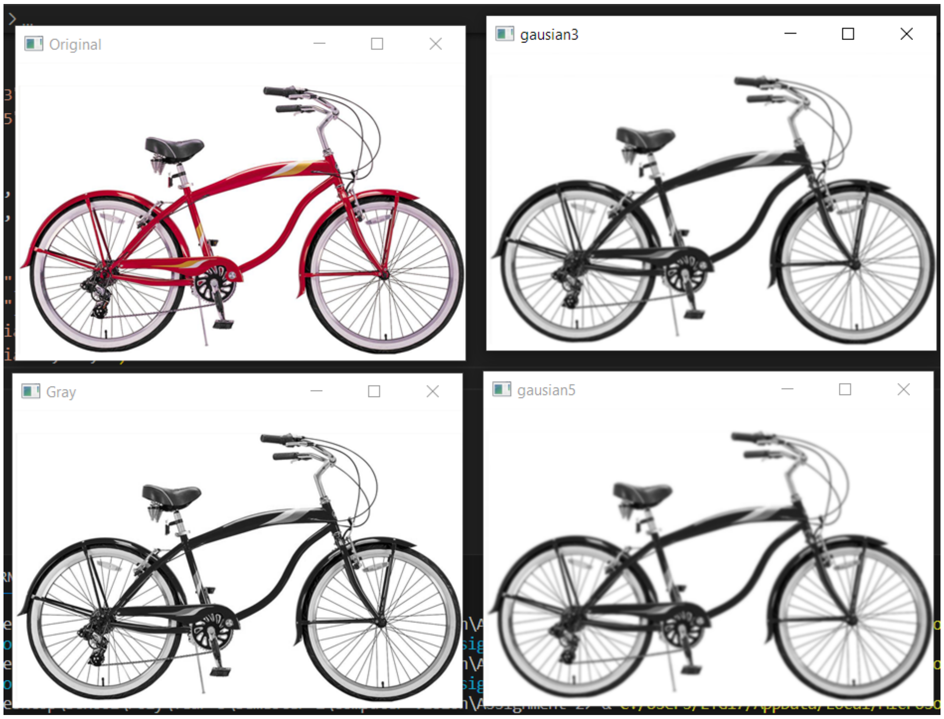


Figure 3.1. Gaussian Filters. Original Image (top left), 3x3 Gaussian Filter (top right), Gray Image (bottom left), 5x5 Gaussian Filter (bottom right)