Hoang Anh (Benjamin) Nguyen
Computer Vision

HW2: Filtering

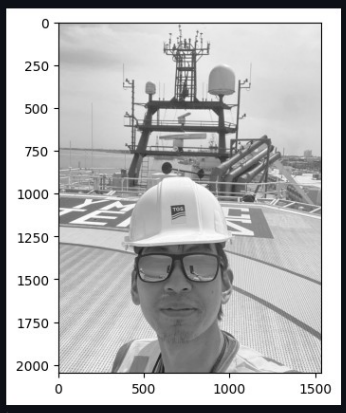
1. Snapshot of the code:

```
[29]  from skimage import io
      from matplotlib import pyplot as plt
      from skimage.color import rgb2gray
      from skimage import util
      import numpy as np

      ✓ 0.0s
```

```
[30]  image_rgb = io.imread('99046aab-a2ae-420f-b837-dec74c8c0636.jpeg')
      image_gray = rgb2gray(image_rgb)
      image_gray_float32 = util.img_as_float32(image_gray)

      ✓ 0.0s
```

```
[31]  plt.imshow(image_gray_float32,cmap=plt.cm.gray)

      ✓ 0.6s
```

```
...   <matplotlib.image.AxesImage at 0x75ee6a87bc50>
```



Filtering

```
[32]  def apply_nxn_filter(image, kernel, stride, padding=True):
          k = kernel.shape[0]

          if padding:
              pad = k // 2
              image_padded = np.pad(image, ((pad, pad), (pad, pad)), mode='constant')
          else:
              image_padded = image

          out_h = (image_padded.shape[0] - k) // stride + 1
          out_w = (image_padded.shape[1] - k) // stride + 1
          output = np.zeros((out_h, out_w))

          # Convolution
          for i in range(0, out_h):
              for j in range(0, out_w):
```

```
    output = np.zeros((out_h, out_w))

    # Convolution
    for i in range(0, out_h):
        for j in range(0, out_w):
            region = image_padded[i*stride:i*stride+k, j*stride:j*stride+k]
            output[i, j] = np.sum(region * kernel)

    return output
```

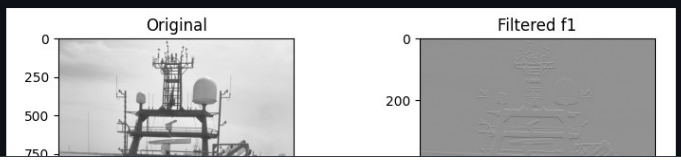
[32] ✓ 0.0s

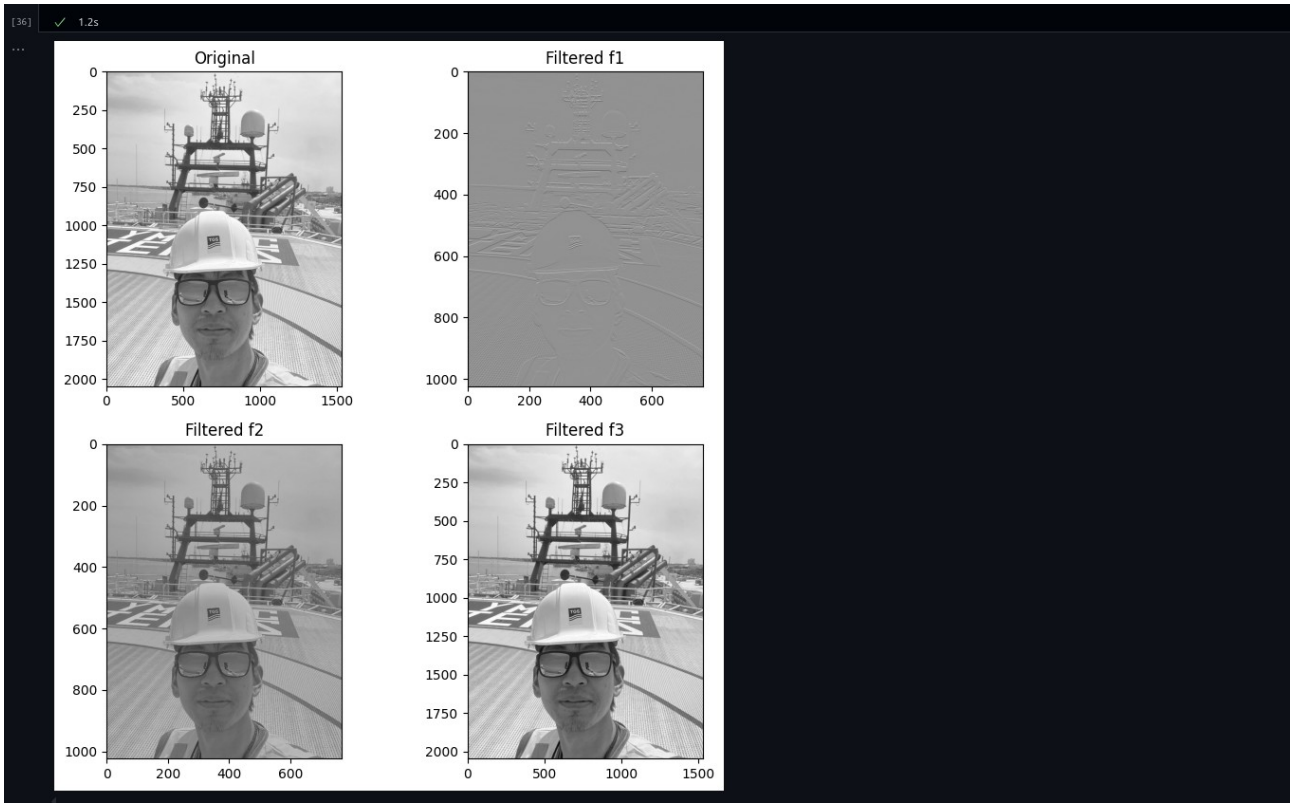
```
# kernel
f1 = np.array([[1,  2,  1],
               [0,  0,  0],
               [-1, -2, -1]])


f2 = np.array([[0, 0, 0],
               [0, 2, 0],
               [0, 0, 0]]) - 1/9 * np.ones((3,3))


f3 = 1/16 * np.array([[1, 2, 1],
                      [2, 4, 2],
                      [1, 2, 1]])

newimage_1 = apply_nxn_filter(image_gray_float32, kernel=f1, stride=2, padding=True)
newimage_2 = apply_nxn_filter(image_gray_float32, kernel=f2, stride=2, padding=True)
newimage_3 = apply_nxn_filter(image_gray_float32, kernel=f3, stride=1, padding=True)
```

[35] ✓ 26.4s

```
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 8))

plt.subplot(2, 2, 1)
plt.imshow(image_gray_float32, cmap=plt.cm.gray)
plt.title("Original")

plt.subplot(2, 2, 2)
plt.imshow(newimage_1, cmap=plt.cm.gray)
plt.title("Filtered f1")

plt.subplot(2, 2, 3)
plt.imshow(newimage_2, cmap=plt.cm.gray)
plt.title("Filtered f2")

plt.subplot(2, 2, 4)
plt.imshow(newimage_3, cmap=plt.cm.gray)
plt.title("Filtered f3")

plt.tight_layout()
plt.show()
```

[36] ✓ 1.2s





2. Explanation

Filter f1 and filter f2 reduce the size of the original image because I used stride = 2.
f1: Sobel filter: in this case, the filter detects horizontal edges.
f2: Removes smooth background, keeps fine details.
f3: Gaussian blur: smooths the image, reduces noise and detail.