```python
# PS3-2 Edge detection
import cv2
import numpy as np
import os

# Sobel Filter Function
def sobel_filter(grayscale_image_sobel):
    sobel_horizontal = 1 / 16 * np.array([[-1, -2, 0, 2, 1], [-2, -4, 0, 4, 2], [-3, -6, 0, 6, 3], [-2, -4, 0, 4, 2], [-1, -2, 0, 2, 1]])
    sobel_vertical = 1 / 16 * np.array([[1, 2, 3, 2, 1], [2, 4, 6, 4, 2], [0, 0, 0, 0, 0], [-2, -4, -6, -4, -2], [-1, -2, -3, -2, -1]])

    edge_horizontal = cv2.filter2D(grayscale_image_sobel, cv2.CV_64F, sobel_horizontal)
    edge_vertical = cv2.filter2D(grayscale_image_sobel, cv2.CV_64F, sobel_vertical)

    edge_magnitude = np.sqrt(edge_horizontal ** 2 + edge_vertical ** 2)
    edge_direction = np.arctan2(edge_vertical, edge_horizontal)

    return edge_magnitude, edge_direction

# Canny Edge Filter Function
def canny_edge_filter(grayscale_image_canny, threshold1, threshold2, aperture_size, l2_gradient):
    aperture_size = max(3, min(aperture_size, 7))
    aperture_size = aperture_size if aperture_size % 2 != 0 else aperture_size - 1

    canny_edge = cv2.Canny(grayscale_image_canny, threshold1, threshold2, apertureSize=aperture_size, L2gradient=l2_gradient)
    negate_canny_edge_image = 255 - canny_edge

    cv2.imshow('Canny Edges',
```

```
27 negate_canny_edge_image)
28
29     return negate_canny_edge_image
30
31 # main script with user input feature
32 user_input = input("Please name your input color
   file: ")
33 file_directory = os.getcwd()
34 image_location = os.path.join(file_directory,
   user_input)
35 if os.path.exists(image_location):
36     print(f"Your '{user_input}' image loaded
   successfully.")
37     input_image = cv2.imread(user_input)
38     cv2.imshow(f"'{user_input}'", input_image)
39     cv2.waitKey(0)
40
41     # Executing Sobel filter function
42     grayscale_image_sobel = cv2.cvtColor(
   input_image, cv2.COLOR_BGR2GRAY)
43     edge_magnitude, edge_direction = sobel_filter(
   grayscale_image_sobel)
44
45     max_edge_magnitude = np.max(edge_magnitude)
46     min_edge_magnitude = np.min(edge_magnitude)
47
48     if max_edge_magnitude != min_edge_magnitude:
49         edge_magnitude_normalized = 255 * (
   edge_magnitude - min_edge_magnitude) / (
50                     max_edge_magnitude -
   min_edge_magnitude)
51     else:
52         edge_magnitude_normalized = edge_magnitude
53
54     edge_magnitude_normalized =
   edge_magnitude_normalized.astype(np.uint8)
55     negate_sobel_image = 255 -
   edge_magnitude_normalized
56
57     # Converting grayscale into binary image
58     if user_input == 'cheerios.png':
```

```
59          threshold_value = 195
60      elif user_input == 'professor.png':
61          threshold_value = 220
62      elif user_input == 'gear.png':
63          threshold_value = 155
64      else:
65          threshold_value = 240
66
67      _, binary_sobel_image = cv2.threshold(
    negate_sobel_image, threshold_value, 255, cv2.
    THRESH_BINARY)
68
69      # Showing and saving Sobel filtered results
70      cv2.imshow(f"'{user_input}'",
    binary_sobel_image)
71      output_image_sobel = user_input.split('.')[0
    ] + '-sobel.' + user_input.split('.')[-1]
72      cv2.imwrite(output_image_sobel,
    binary_sobel_image)
73      cv2.waitKey(0)
74
75      # Executing Canny edge filter function
76      grayscale_image_canny = cv2.cvtColor(
    input_image, cv2.COLOR_BGR2GRAY)
77
78      # Canny edge GUI
79      cv2.namedWindow('Canny Edge GUI')
80      cv2.createTrackbar('Threshold1', 'Canny Edge
    GUI', 0, 255, lambda x: None)
81      cv2.createTrackbar('Threshold2', 'Canny Edge
    GUI', 0, 255, lambda x: None)
82      cv2.createTrackbar('Aperture Size', 'Canny Edge
     GUI',3, 7, lambda x: None)
83      cv2.createTrackbar('L2 Gradient', 'Canny Edge
    GUI', 0, 1, lambda x: None)
84
85      while True:
86          threshold1 = cv2.getTrackbarPos('Threshold1
    ', 'Canny Edge GUI')
87          threshold2 = cv2.getTrackbarPos('Threshold2
    ', 'Canny Edge GUI')
```

```
 88          aperture_size = cv2.getTrackbarPos('Aperture Size', 'Canny Edge GUI')
 89          l2_gradient = cv2.getTrackbarPos('L2 Gradient', 'Canny Edge GUI')
 90
 91          canny_edge_result = canny_edge_filter(grayscale_image_canny, threshold1, threshold2, aperture_size, l2_gradient)
 92          negate_canny_edge_image = 255 - canny_edge_result
 93          cv2.imshow('Canny Edges', canny_edge_result)
 94
 95          key = cv2.waitKey(1) & 0xFF
 96          if key == ord(' '):  # If the space key is pressed
 97              output_image_canny = user_input.split('.')[0] + '-canny.' + user_input.split('.')[-1]
 98              cv2.imwrite(output_image_canny, canny_edge_result)
 99              print(f"Canny edge filter image saved as: {output_image_canny}")
100              break
101      cv2.destroyAllWindows()
102
103  else:
104      print(f"Error: unable to load your input image.\nPlease make sure '{user_input}' is in the correct directory.")
105      exit()
```