Homework-4

Q1:

A screenshot of a cell phone

Description automatically generated

1. Compute g1, h1, g2, where:

1) g1 = f1 ⊗ g

f1’ =

|  |  |  |
| --- | --- | --- |
| 2 | 0 | 1 |

g1 = f1 ⊗ g = f1’ ∗ g =

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2 | 0+2 | 1+0+2 | 1+0+2 | 1+0+2 | 1+0 | 1 |
| 0 | 0+2 | 0+0+4 | 1+0+6 | 2+0+8 | 3+0 | 4 |
| 0 | 0+0 | 0+0+0 | 0+0+0 | 0+0+0 | 0+0 | 0 |
| 4 | 0+4 | 2+0+4 | 2+0+4 | 2+0+4 | 2+0 | 2 |

=

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2 | 2 | 3 | 3 | 3 | 1 | 1 |
| 0 | 2 | 4 | 7 | 10 | 3 | 4 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 4 | 6 | 6 | 6 | 2 | 2 |

2) h1 = f1 ∗ g

h1 = f1 ∗ g =

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 0+1 | 2+0+1 | 2+0+1 | 2+0+1 | 2+0 | 2 |
| 0 | 0+1 | 0+0+2 | 2+0+3 | 4+0+4 | 6+0 | 8 |
| 0 | 0+0 | 0+0+0 | 0+0+0 | 0+0+0 | 0+0 | 0 |
| 2 | 0+2 | 4+0+2 | 4+0+2 | 4+0+2 | 4+0 | 4 |

=

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 3 | 3 | 3 | 2 | 2 |
| 0 | 1 | 2 | 5 | 8 | 6 | 8 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 2 | 6 | 6 | 6 | 4 | 4 |

3) g2 = f2 ⊗ g1

f2’ =

|  |  |
| --- | --- |
| -1 | 0 |
| 1 | 1 |

g1’ =

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2 | 2 | 3 | 3 | 3 | 1 | 1 |
| 0 | 2 | 4 | 7 | 10 | 3 | 4 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 4 | 6 | 6 | 6 | 2 | 2 |

g2 = f2 ⊗ g1 = f2’ ∗ g1 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -2 | 0-2 | 0-3 | 0-3 | 0-3 | 0-1 | 0-1 | 0 |
| 2+0 | 2+2+0-2 | 2+3+0-4 | 3+3+0-7 | 3+3+0-10 | 3+1+0-3 | 1+1+0-4 | 1+0 |
| 0+0 | 0+2+0+0 | 2+4+0+0 | 4+7+0+0 | 7+10+0+0 | 10+3+0+0 | 3+4+0+0 | 4+0 |
| 0-4 | 0+0+0-4 | 0+0+0-6 | 0+0+0-6 | 0+0+0-6 | 0+0+0-2 | 0+0+0-2 | 0+0 |
| 4 | 4+4 | 4+6 | 6+6 | 6+6 | 6+2 | 2+2 | 2 |

=

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -2 | -2 | -3 | -3 | -3 | -1 | -1 | 0 |
| 2 | 2 | 1 | -1 | -4 | 1 | -2 | 1 |
| 0 | 2 | 6 | 11 | 17 | 13 | 7 | 4 |
| -4 | -4 | -6 | -6 | -6 | -2 | -2 | 0 |
| 4 | 8 | 10 | 12 | 12 | 8 | 4 | 2 |

2. Compute a mask f such that:

g2 = f ⊗ g

From 1 we can see:

g2 = f2 ⊗ g1 = f2 ⊗ (f1 ⊗ g) = f2 ⊗ (f1’ ∗ g) = f2’ ∗ (f1’ ∗ g) = (f2’ ∗ f1’) ∗ g = (f2’ ∗ f1’)’ ⊗ g

∴f = (f2’ ∗ f1’)’

f1’ =

|  |  |  |
| --- | --- | --- |
| 2 | 0 | 1 |

f2’ =

|  |  |
| --- | --- |
| -1 | 0 |
| 1 | 1 |

f2’ ∗ f1’ =

|  |  |  |  |
| --- | --- | --- | --- |
| -2 | 0+0 | -1+0 | 0 |
| 2 | 0+2 | 1+0 | 1 |

=

|  |  |  |  |
| --- | --- | --- | --- |
| -2 | 0 | -1 | 0 |
| 2 | 2 | 1 | 1 |

f = (f2’ ∗ f1’)’ =

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 1 | 2 | 2 |
| 0 | -1 | 0 | -2 |

Q2:

Write a program that generate a gray-level image of 200 rows and 500 columns. Associate the value x = (255 \* j / 499.0) with the pixel at row i and column j. Convert it to a gray level by rounding. How many distinct values are you able to detect in the image?

(In your answer, you need to submit only the number of distinct values that you are observing. Not the source code.)

Solution:

Interval = 255 / 499 < 1

∴ The difference between adjacent columns does not exceed 1.

∴ Every integer from the StartValue to the Endvalue appears.

J ∈ [0, 499]

X ∈ [0, 255]

∴ There are (255 – 0) + 1 = 256 values I can detect in the image. (Result is checked in code.)

Q3:

Write a program that creates the negative image for a given image. Your program should do the following: • Read an image.

• Display the input image.

• If the image is color, convert it into a gray level image.

• Display the gray level image.

• Compute the “negative” of the gray level image.

• Display the negative image.

• Write the negative image.

You may want to write your program by modifying the example program “GrayImages.py”.

Code:

**import** cv2

**import** sys

**import** numpy **as** np # arrays

# Description:

# Write a program that creates the negative image for a given image. Your program should do the following: • Read an image.

# • Display the input image.

# • If the image is color, convert it into a gray level image.

# • Display the gray level image.

# • Compute the “negative” of the gray level image.

# • Display the negative image.

# • Write the negative image.

# You may want to write your program by modifying the example program “GrayImages.py”.

# argv[1] == target image file.

**if**(len(sys.argv) != 3) :

print(sys.argv[0], "takes 2 arguments. Not ", len(sys.argv)-1)

sys.exit()

name\_input = sys.argv[1]

name\_output = sys.argv[2]

image\_input = cv2.imread(name\_input, cv2.IMREAD\_UNCHANGED);

**if**(image\_input **is** **None**) :

print(sys.argv[0], "Failed to read image from ", name\_input)

sys.exit()

cv2.imshow('original image', image\_input);# Display the input image

rank = len(image\_input.shape)

**if**(rank == 2) :

gray\_image = image\_input

**elif**(rank == 3) :

gray\_image = cv2.cvtColor(image\_input, cv2.COLOR\_BGR2GRAY)# If the image is color, convert it into a gray level image.

**else** :

print(sys.argv[0], "Can't handle unusual image ", name\_input)

sys.exit()

cv2.imshow('gray image', gray\_image);# Display the gray level image.

rows, cols = gray\_image.shape

image\_negative = np.zeros([rows, cols], dtype=np.uint8)

# Compute the “negative” of the gray level image.

**for** i **in** range(0, rows) :

**for** j **in** range(0, cols) :

image\_negative[i,j] = 255 - gray\_image[i,j]

cv2.imshow('negative image', image\_negative);# Display the negative image.

cv2.imwrite(name\_output, image\_negative);# Write the negative image.

# wait for key to exit

cv2.waitKey(0)

cv2.destroyAllWindows()