

Professor: Tiago B. A. de Carvalho.

Processamento Digital de Imagens: Operações Básicas

Atenção: a resposta de cada questão dever estar acompanhada de:

- ^ Código-fonte com a implementação da resposta;
- ^ Imagens de saída;
- ^ Um texto de análise dos resultados.

1. (20 pontos) Utilizando as imagens `lena gray 512.png` e `mandril gray.png`, exibidas na Figura 1, gere novas imagens a partir da soma ponderada dos pixels de cada imagem. Seja $f(x, y)$ e $g(x, y)$ as imagens de entrada, a nova imagem é $h(x, y) = \alpha f(x, y) + (1-\alpha)g(x, y)$, $0 \leq \alpha \leq 1$; a nova imagem é dita uma interpolação linear das outras duas. Gere as imagens para os seguintes valores de α : 0,25; 0,50; e 0,75. O que pode-se notar das imagens geradas?



Figura 1: lena gray 512.png e mandril gray.png.

```
import numpy as np
import cv2
0.0s
Python

OpenCV: detecção e reconhecimento de faces, identificação de objetos, classificação de ações humanas em vídeos, rastreamento de movimentos de objetos, dentre outros.

O método cv2.imread() carrega uma imagem do arquivo especificado. Se a imagem não puder ser lida

f=cv2.imread('/home/laisy/Documentos/RP/RP-2022.1/semana 9/imagens/lena_gray_512.png', cv2.IMREAD_GRAYSCALE)
g=cv2.imread('/home/laisy/Documentos/RP/RP-2022.1/semana 9/imagens/mandril_gray.png', cv2.IMREAD_GRAYSCALE)
0.1s
Python

def sum_img(img_1, img_2, n):
    result = input_type_conversion(img_1)
    result = sum_of_images(result, img_2, n)
    result = correction(result)

    return result
```

```

def correction(img):
    maxi = 0
    mini = 255

    for x in range(len(img)):
        for y in range(len(img[x])):
            maxi = max(maxi, img[x][y])
            mini = min(mini, img[x][y])

    for k in range(len(img)):
        for z in range(len(img[k])):
            img[k][z] = np.uint8(255 * (img[k][z]-mini)/(maxi-mini))

    return np.array(img)

```

■ Correção da adição e subtração

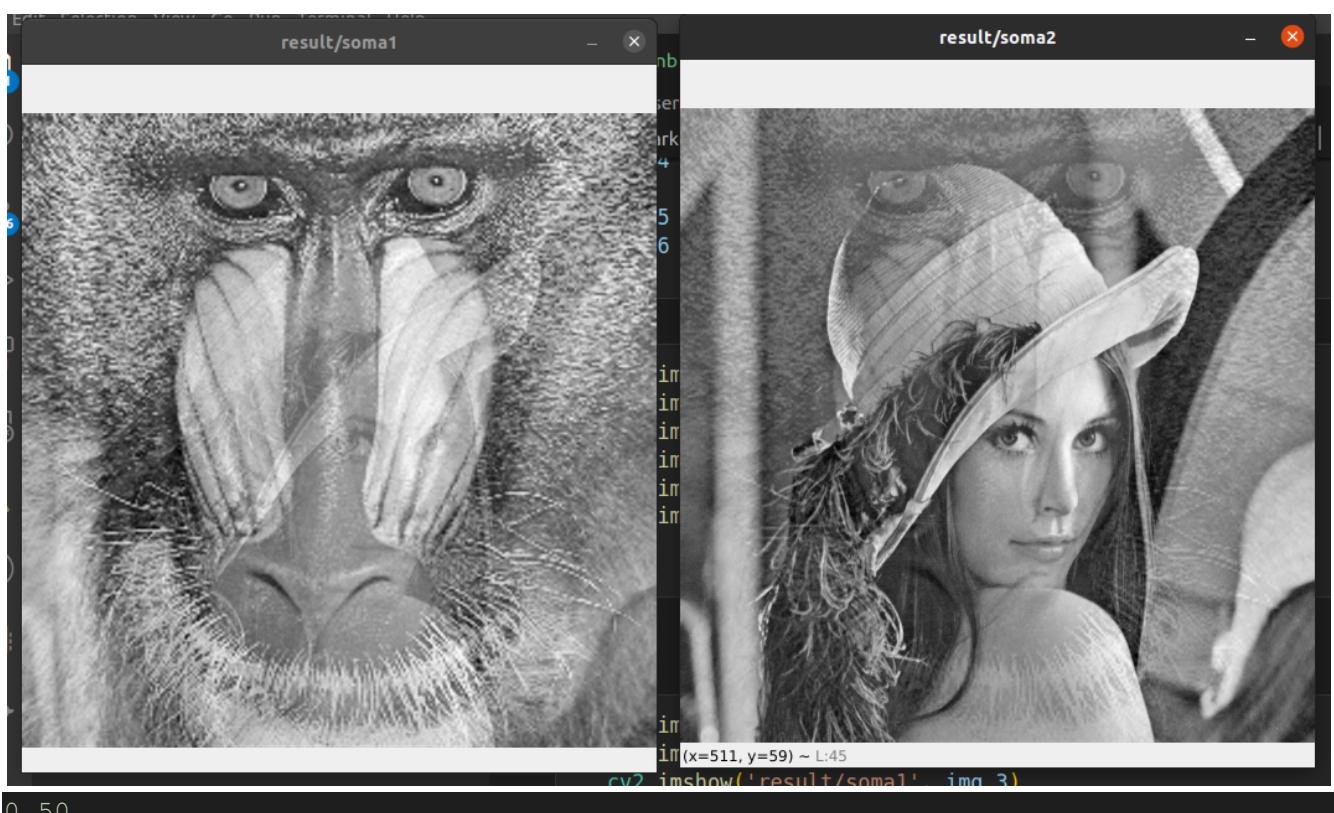
- $f' = 255 * (f-\min)/(max-\min)$

```

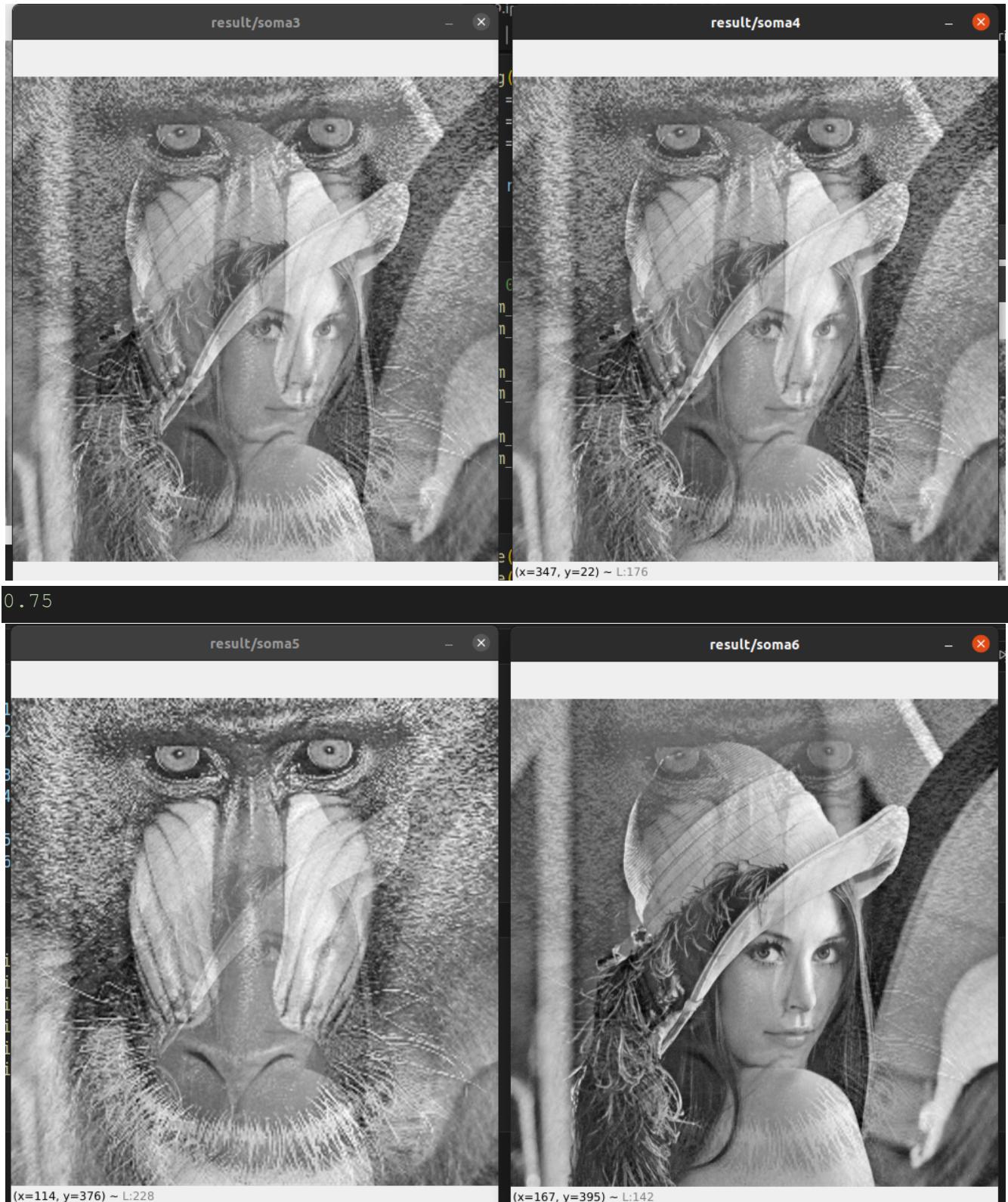
def sum_of_images(array_aux, g, n):
    for p in range(len(array_aux)):
        for u in range(len(array_aux[p])):
            array_aux[p][u] = (n*array_aux[p][u]) + int((1-n)*g[p][u])
    return array_aux

```

0.25



0.50



2. (20 pontos) O que se pode notar pela diferença entre as imagens Fig0228(a)(angiography mask image) e Fig0228(b)(angiography live image) , descritas na Figura 2? Compare os resultados subtraindo a primeira imagem da segunda e subtraindo a segunda imagem da primeira. Obs.: a subtração de imagens pode causar alguns problemas em relação ao valor mínimo e máximo das intensidades do pixel que precisam ser corrigidos antes da visualização.

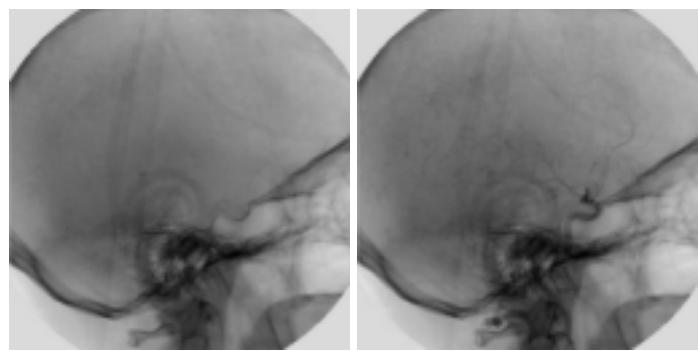


Figura 2: Fig0228(a)(angiography mask image) e Fig0228(b)(angiography live image).

```

def image_subtraction(array_aux, img2):
    for p in range(len(array_aux)):
        for u in range(len(array_aux[p])):
            array_aux[p][u] = (array_aux[p][u]) - int(img2[p][u])
    return array_aux
✓ 0.0s

def sub_img(img_1, img_2):
    result = input_type_conversion(img_1)
    result = image_subtraction(result, img_2)
    result = correction(result)

    return result
✓ 0.0s

```

The code defines two functions: `image_subtraction` and `sub_img`. The `image_subtraction` function takes two arrays, `array_aux` and `img2`, and performs element-wise subtraction. The `sub_img` function uses `image_subtraction` along with `input_type_conversion` and `correction` functions to subtract `img_2` from `img_1`.

Below the code, two windows are shown: `result_Q2/subtracao1` and `result_Q2/subtracao2`. The left window shows the original angiography mask image. The right window shows the result of the subtraction, where the vessels appear brighter against a darker background.

3. (20 pontos) Realize a correção de iluminação pela divisão da imagem

Fig0229(a)(tungsten lament shaded).png pela imagem Fig0229(b)(tungsten sensor shading).png , exibidas na Figura 3. Pode-se perceber que a área escurecida no canto inferior direito foi corrigida?

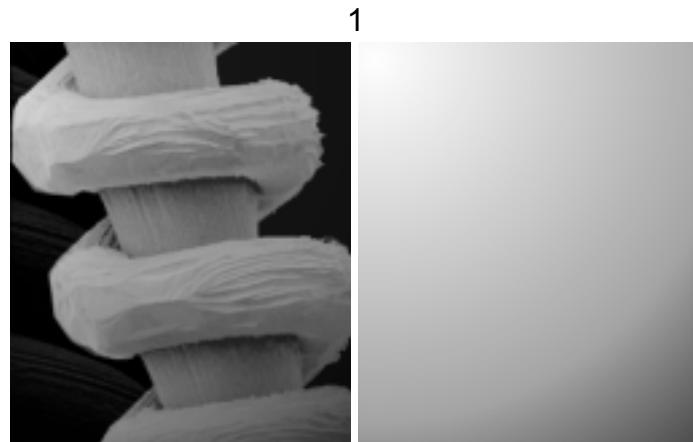
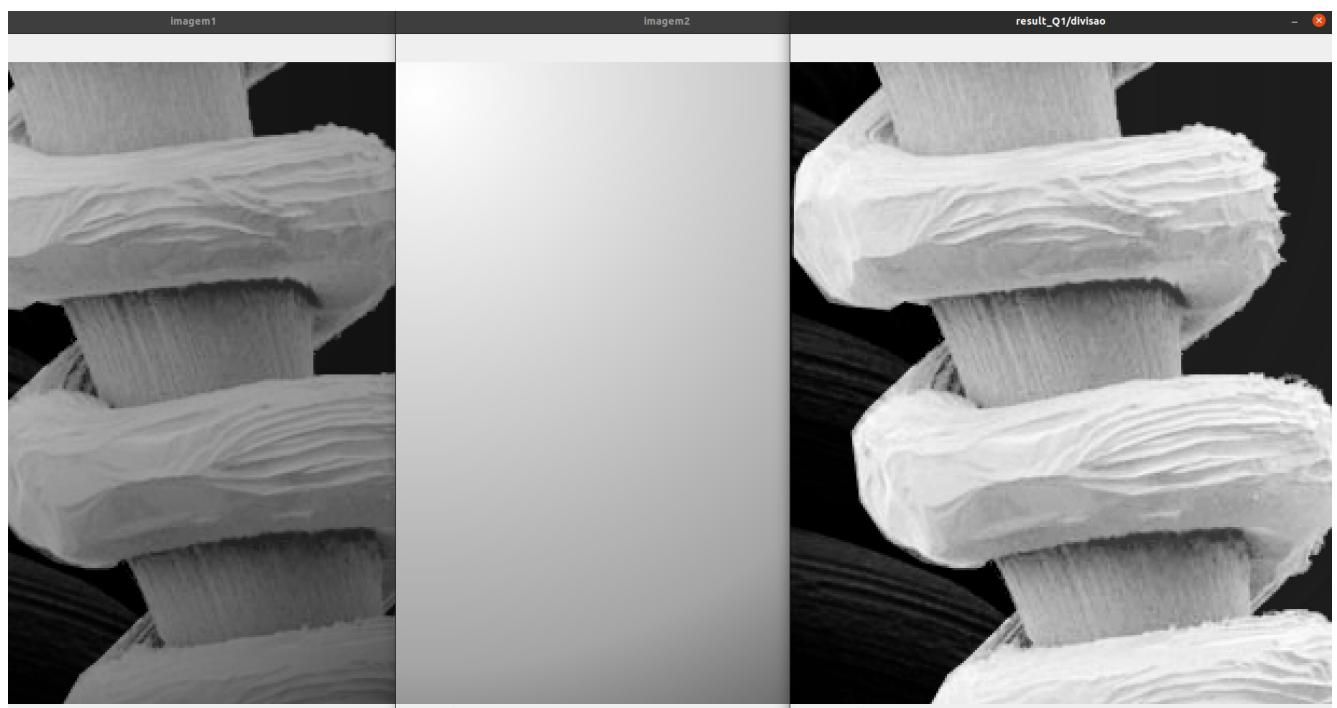


Figura 3: Fig0229(a)(tungsten lament shaded) e Fig0229(b)(tungsten sensor shading).



```
def image_division(array_aux, img2):
    for p in range(len(array_aux)):
        for u in range(len(array_aux[p])):
            if img2[p][u] == 0:
                array_aux[p][u] = (array_aux[p][u])/(int(img2[p][u])+1)
            else:
                array_aux[p][u] = (array_aux[p][u])/(int(img2[p][u]))
```

✓ 0.0s

4. (20 pontos) Qual o resultado do produto das imagens Fig0230(a)(dental xray) e Fig0230(b) (dental xray mask) , exibidas na Figura 4? Experimente mudar a máscara e obter outros resultados.

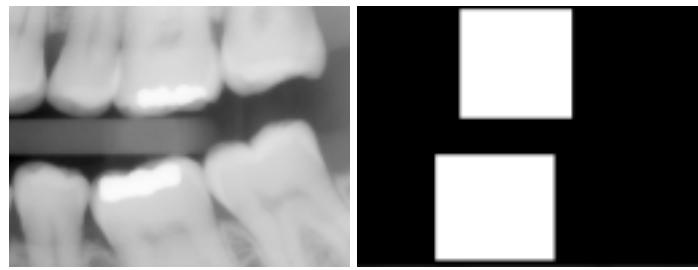
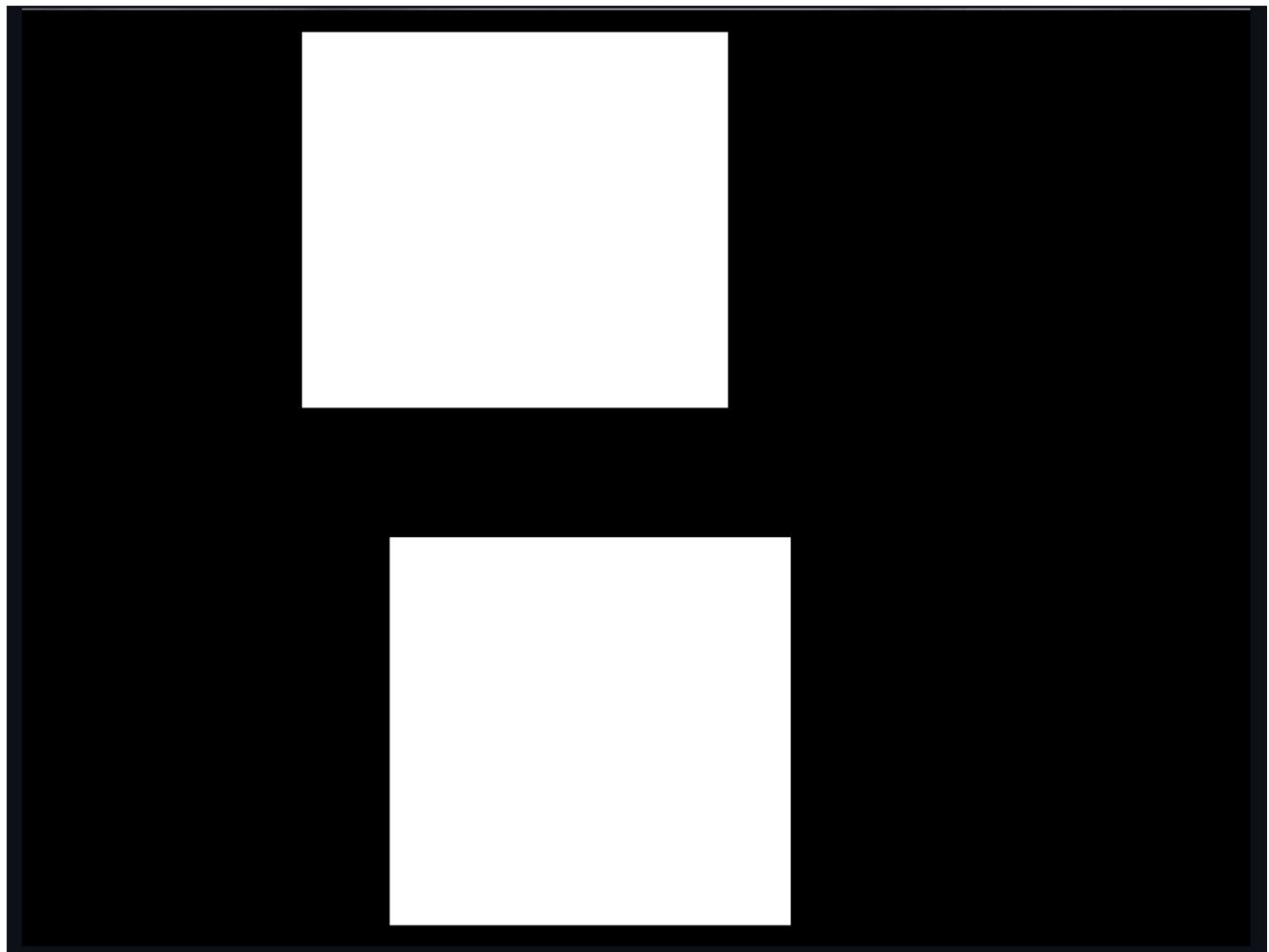
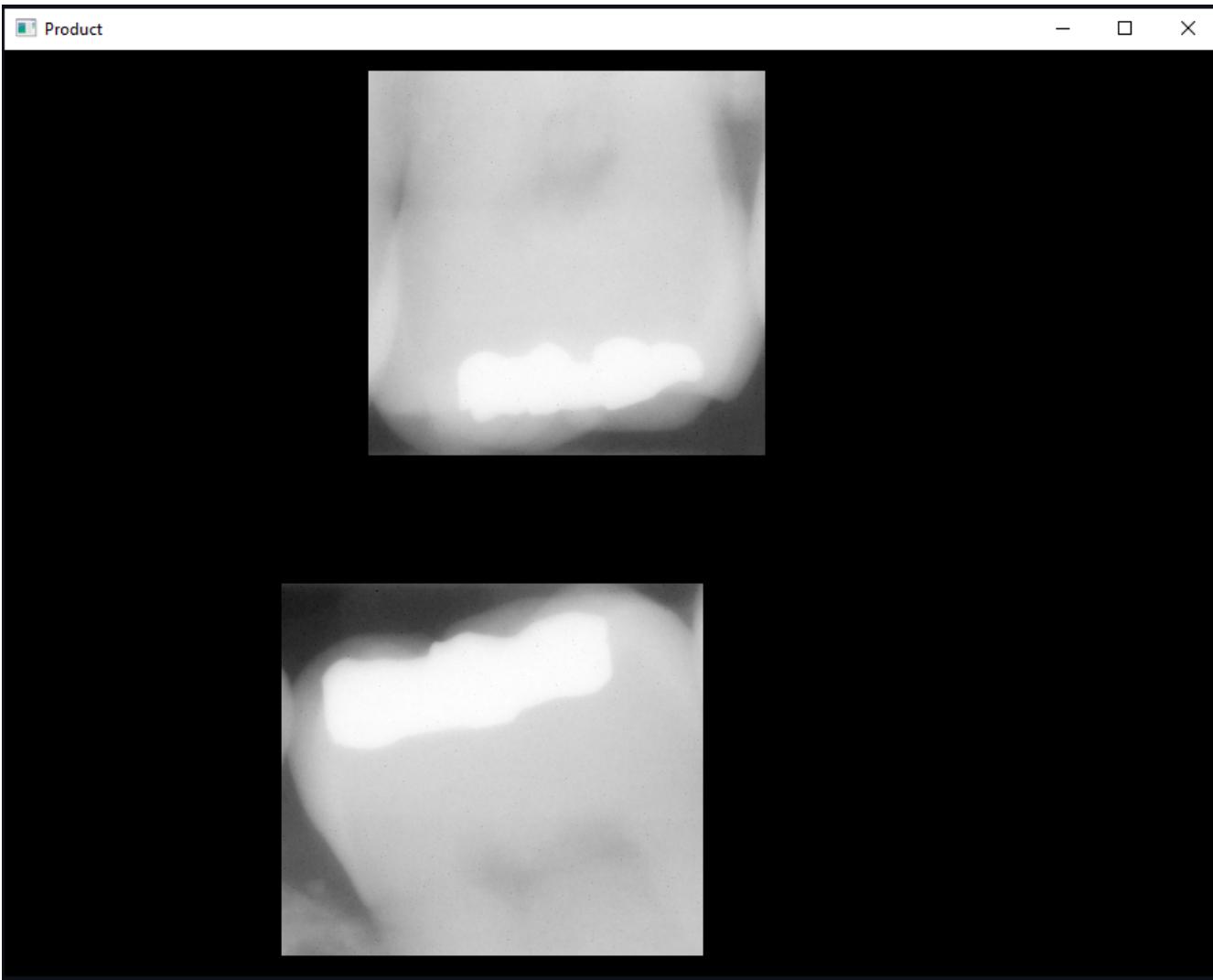
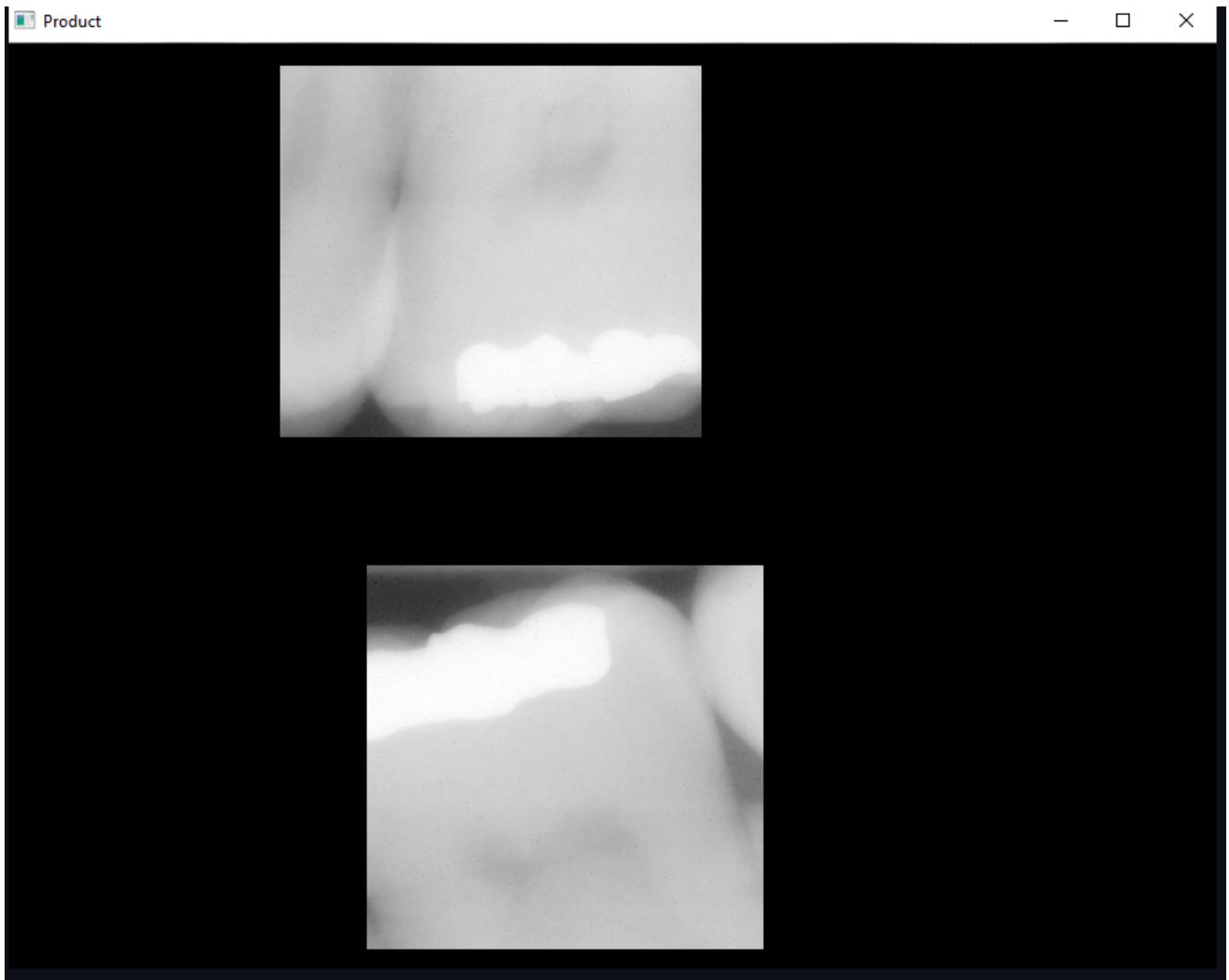


Figura 4: Fig0230(a)(dental xray) e Fig0230(b)(dental xray mask).

```
1 import cv2
2 import numpy as np
3
4 img1 = cv2.imread(r'c:\Users\David\Desktop\RP-2022.1\semana 9\Imagens\Fig0230(a)(dental_xray).png', cv2.IMREAD_GRAYSCALE) / 255
5 img2 = cv2.imread(r'c:\Users\David\Desktop\RP-2022.1\semana 9\Imagens\Fig0230(b)(dental_xray_mask)2.png', cv2.IMREAD_GRAYSCALE) / 255
6
7 resultado = np.multiply(img1, img2)
8 cv2.imshow('Product', resultado)
9 cv2.waitKey(0)
10 cv2.destroyAllWindows()
```







5. (20 pontos) A Figura 5 mostra algumas das 100 imagens de Lena nas quais foram adicionado ruído.

Somatório das imagens

```
def sum_of_images(qtd, array_aux):
    for x in range(qtd):
        img2=cv2.imread('/home/laisy/Documentos/RP/RP-2022.1/semana 9/ruido/lena'+str(x+2)+'.png', cv2.IMREAD_GRAYSCALE)
        for p in range(len(array_aux)):
            for u in range(len(array_aux[p])):
                array_aux[p][u] = (array_aux[p][u]) + int(img2[p][u])
    return array_aux
```

- (a) Gere uma única imagem através da média destas 100.

somatório de 0 a 100

```
average_100 = sum_of_images(99, array_aux)

average_100 = correction(average_100)

✓ 16.3s

cv2.imwrite('result_Q5/media100.png', average_100)

✓ 0.0s
```

(b) Como ca a imagem gerada a partir da média de apenas 10 destas?

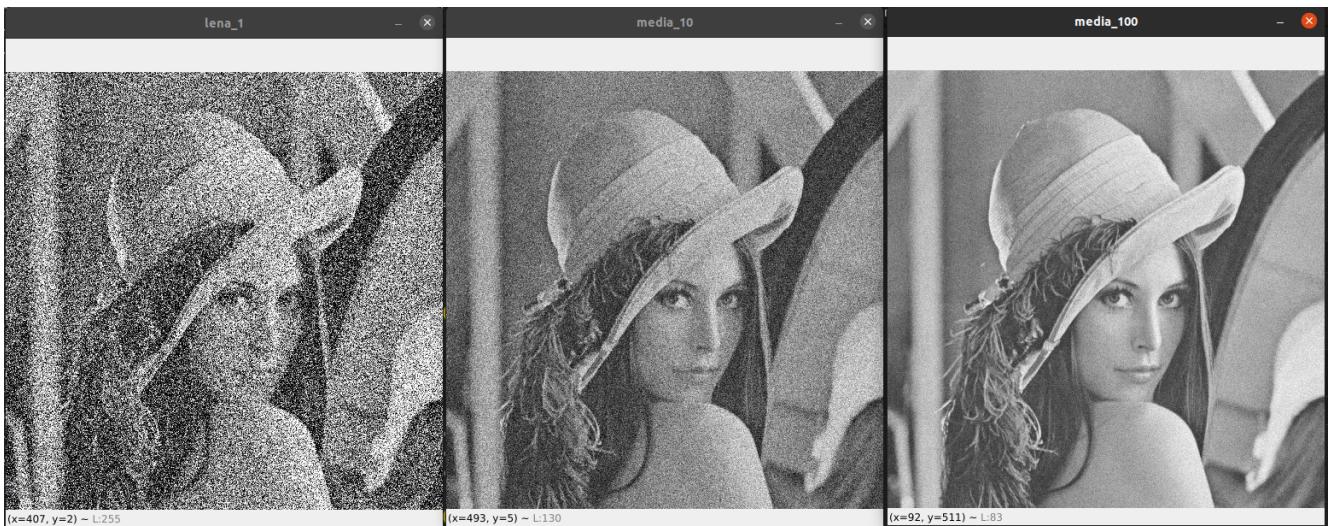
```
average_10 = sum_of_images(9, array_aux)

average_10 = correction(average_10)

✓ 2.9s

cv2.imwrite('result_Q5/media10.png', average_10)

✓ 0.0s
```



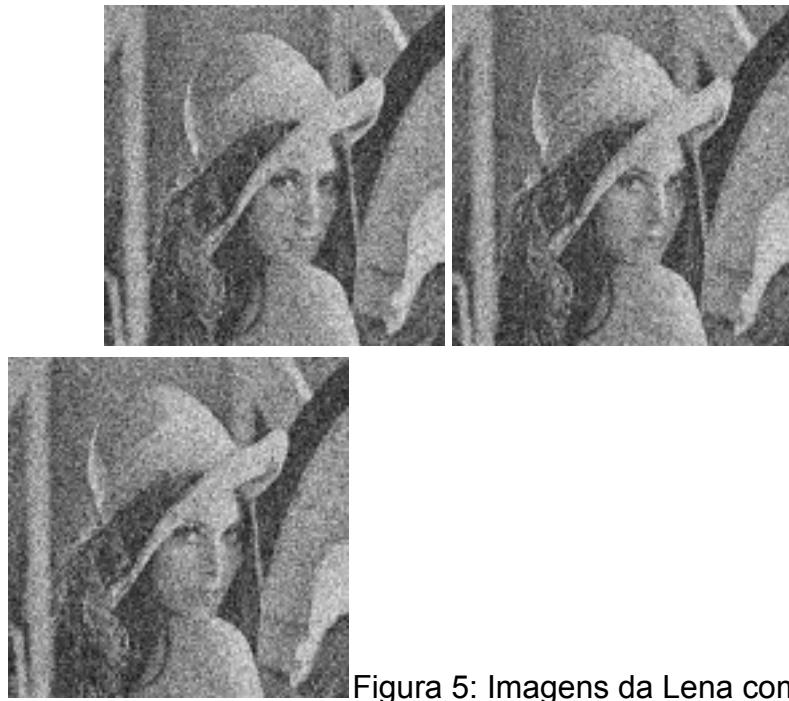


Figura 5: Imagens da Lena com ruído.