INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA SÃO PAULO Campus Birigui	**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA Campus Birigui** **Bacharelado em Engenharia de Computação**	
Disciplina: Processamento Digital de Imagem	Relatório – Transformações básicas	
Professor: Murilo Varges		**Data:** 21/08/2023
Nome do Aluno: Raul Prado Dantas		**Prontuário:** BI3007669

Operação Ponto A Ponto

Birigui 2023

Operações-ponto-a-ponto

August 29, 2023

[94]:
```
import numpy as np
from PIL import Image
from numpy import asarray
import matplotlib.pyplot as plt
from matplotlib import pyplot as plt
```

1 [OPERAÇÕES PONTO A PONTO]

1.1 Calcular o negativo das imagens

1.2 Diminuir pela metade a intensidade dos pixels

1.3 Incluir 4 quadrados brancos 10 x 10 pixels em cada canto das imagens

1.4 Incluir 1 quadrado preto 15X15 no centro das imagens

1.4.1 print_final_result() :: void

função responsável por mostrar o resultado final da imagem através da biblioteca matplotlib e aplicar as transformações geométricas na imagem original.

[95]:
```
def print_final_result(img_path1, img_path2, img_path3, title, function):

    imgLena = Image.open(img_path1)
    f_imgLena = asarray(imgLena)
    imgCameraman = Image.open(img_path2)
    f_imgCameraman = asarray(imgCameraman)
    imgHouse = Image.open(img_path3)
    f_imgHouse = asarray(imgHouse)

    print(title)

    plt1 = plt.subplot(1,3,1)
    plt2 = plt.subplot(1,3,2)
    plt3 = plt.subplot(1,3,3)

    plt1.set_title('Lena')
    plt2.set_title('Cameraman')
    plt3.set_title('House')
```

```
    if(title.find('Exercicio A') != -1):
        plt1.imshow(function(f_imgLena), cmap='gray')
        plt2.imshow(function(f_imgCameraman), cmap='gray', vmin=0, vmax=255)
        plt3.imshow(function(f_imgHouse), cmap='gray', vmin=0, vmax=255)
    else:
        plt1.imshow(function(f_imgLena), cmap='gray')
        plt2.imshow(function(f_imgCameraman), cmap='gray', vmin=0, vmax=255)
        plt3.imshow(function(f_imgHouse), cmap='gray', vmin=0, vmax=255)

    plt.show()
    plt.figure()
    print()
```

1.4.2 negative_image(img) :: img

função responsável por aplicar o negativo na imagem original.

[96]:
```
def negativeImg(img):
    img = asarray(img)
    img = 255 - img
    return img
```

1.4.3 half_intensity(img) :: img

função responsável por aplicar a metade da intensidade na imagem original.

[97]:
```
def halfIntesity(img):
    img = img/2
    return img
```

1.4.4 white_squares(img) :: img

função responsável por aplicar os quadrados brancos nos cantos da imagem original.

[98]:
```
#quadrado branco de 50x50 em cada canto da imagem
def whiteSquare(img):
    img = asarray(img)
    img[0:50,0:50] = 255
    img[0:50, img.shape[1]-50:img.shape[1]] = 255
    img[img.shape[0]-50:img.shape[0], 0:50] = 255
    img[img.shape[0]-50:img.shape[0], img.shape[1]-50:img.shape[1]] = 255

    return img
```

1.4.5 black_squares(img) :: img

função responsável por aplicar o quadrado preto no centro da imagem original.

[99]:
```
# Incluir 1 quadrado preto 50X50 no centro das imagens
def blackSquare(img):
    img = asarray(img)
    img[int(img.shape[0]/2)-25:int(img.shape[0]/2)+25, int(img.shape[1]/2)-25:
 ↪int(img.shape[1]/2)+25] = 0
    return img
```

1.4.6 exercicioA() :: void

função responsável por chamar as funções que aplicam as transformações geométricas na imagem original.

[100]:
```
# Calcular o negativo das imagens
def exercicioA():
    print("Exercicio A: Negativo da imagem")

    print_final_result('lena_gray_512.tif', 'cameraman.tif', 'house.tif',␣
 ↪'Imagens originais', lambda x: x)

    print_final_result('lena_gray_512.tif', 'cameraman.tif', 'house.tif',␣
 ↪'Imagens negativas', negativeImg)
```

1.4.7 exercicio A B C D () :: void

funções responsáveis por chamar as funções que aplicam as transformações de ponto a ponto na imagem original.

[101]:
```
# Diminuir pela metade a intensidade dos pixels
def exercicioB():
    print("Exercicio B: Diminuir pela metade intensidade dos pixels")
    print()
    print_final_result('lena_gray_512.tif', 'cameraman.tif', 'house.tif',␣
 ↪'Imagens originais', lambda x: x)

    print_final_result('lena_gray_512.tif', 'cameraman.tif', 'house.tif',␣
 ↪'Imagens com metade da intensidade', halfIntesity)
```

[102]:
```
# Incluir 4 quadrados brancos 10 x 10 pixels em cada canto das imagens
def exercicioC():
    print("Exercicio C: Incluir 4 quadrados brancos 50 x 50 pixels em cada␣
 ↪canto das imagens")
    print()
    print_final_result('lena_gray_512.tif', 'cameraman.tif', 'house.tif',␣
 ↪'Imagens originais', lambda x: x)

    print_final_result('lena_gray_512.tif', 'cameraman.tif', 'house.tif',␣
 ↪'Imagens com quadrados brancos', whiteSquare)
```

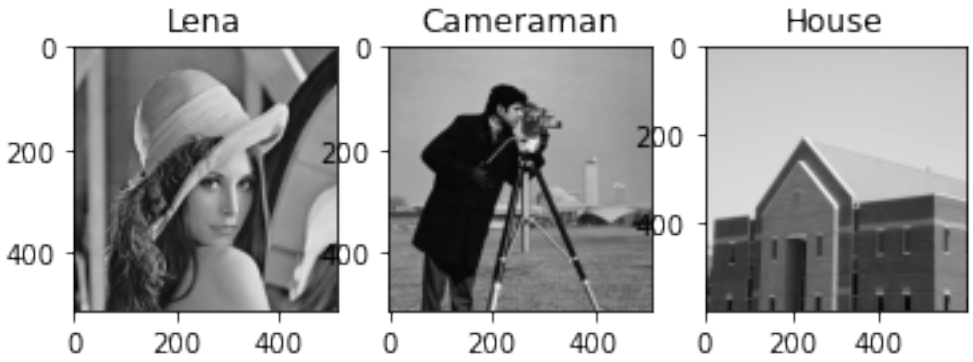
[103]:
```
# Incluir 1 quadrado preto 15X15 no centro das imagens
def exercicioD():
    print("Exercicio D: Incluir 1 quadrado preto 15X15 no centro das imagens")
    print()
    print_final_result('lena_gray_512.tif', 'cameraman.tif', 'house.tif',␣
 ↪'Imagens originais', lambda x: x)

    print_final_result('lena_gray_512.tif', 'cameraman.tif', 'house.tif',␣
 ↪'Imagens com quadrado preto no centro', blackSquare)
```

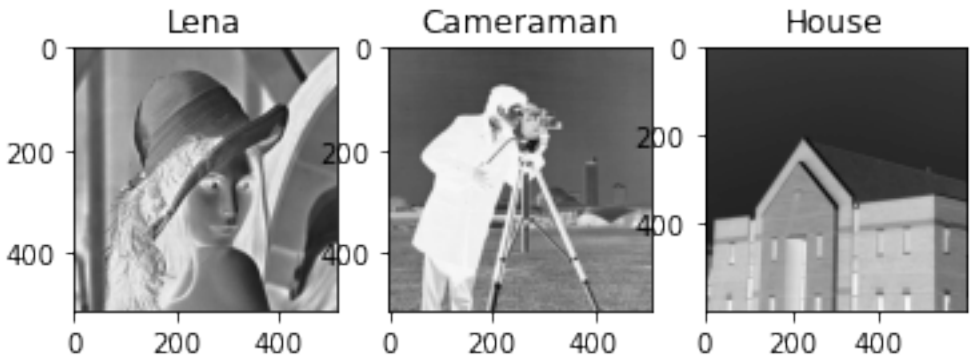
[104]:
```
def main():
    exercicioA()
    exercicioB()
    exercicioC()
    exercicioD()
```

[105]:
```
if __name__ == "__main__":
    main()
```

```
Exercicio A: Negativo da imagem
Imagens originais
```

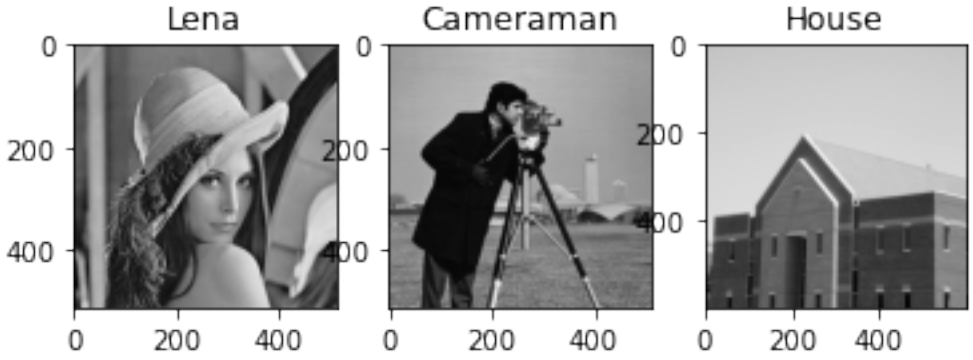



```
Imagens negativas
```

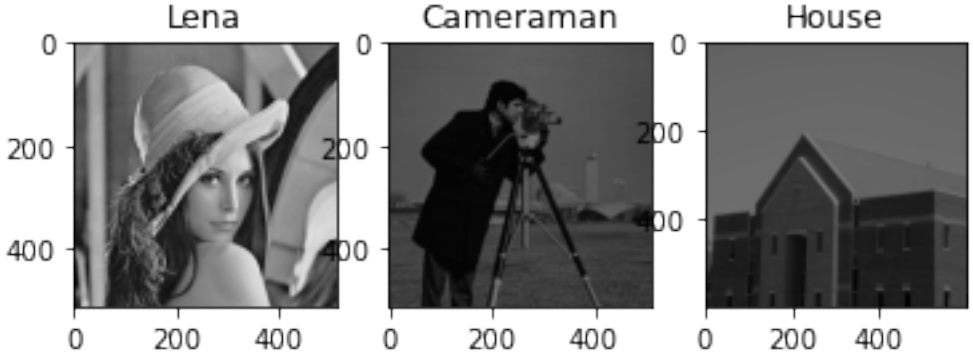


Exercicio B: Diminuir pela metade intensidade dos pixels

Imagens originais

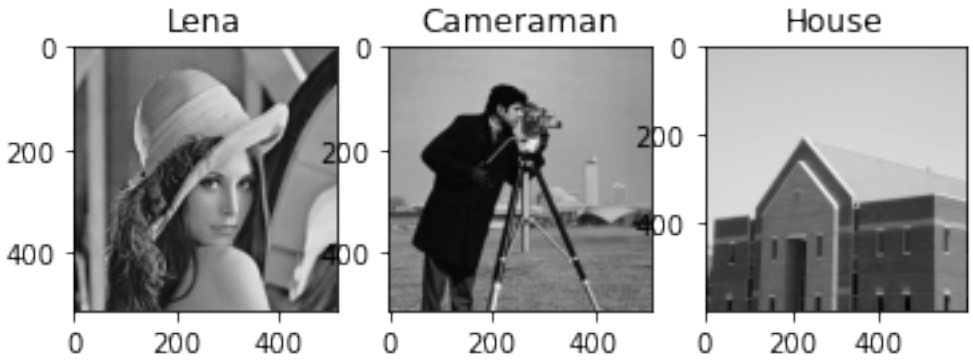


Imagens com metade da intensidade

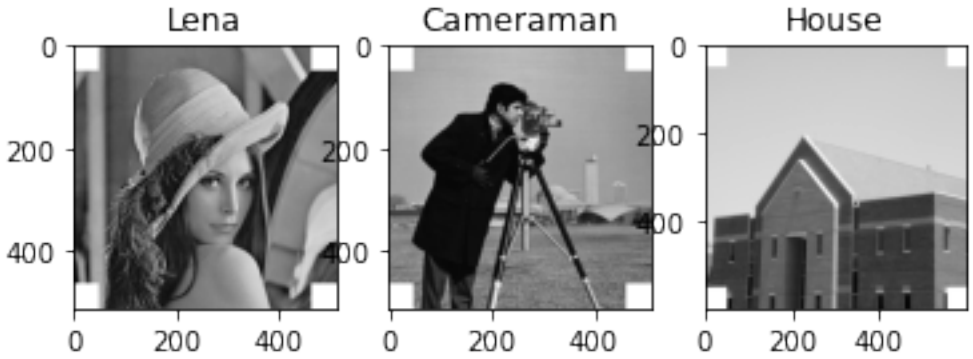


Exercicio C: Incluir 4 quadrados brancos 50 x 50 pixels em cada canto das imagens

Imagens originais

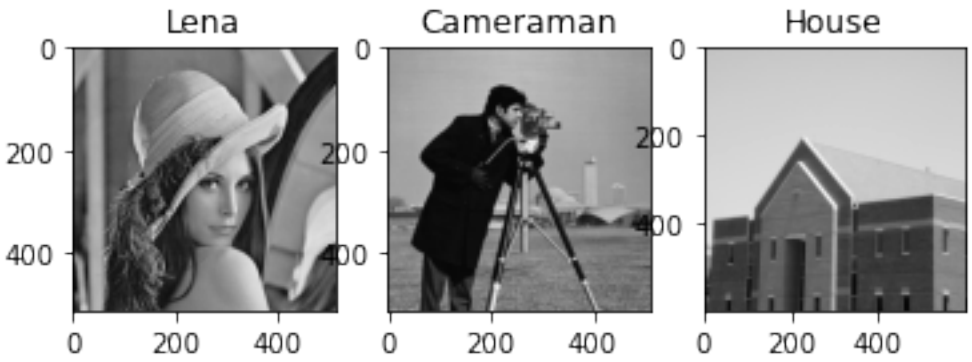


Imagens com quadrados brancos

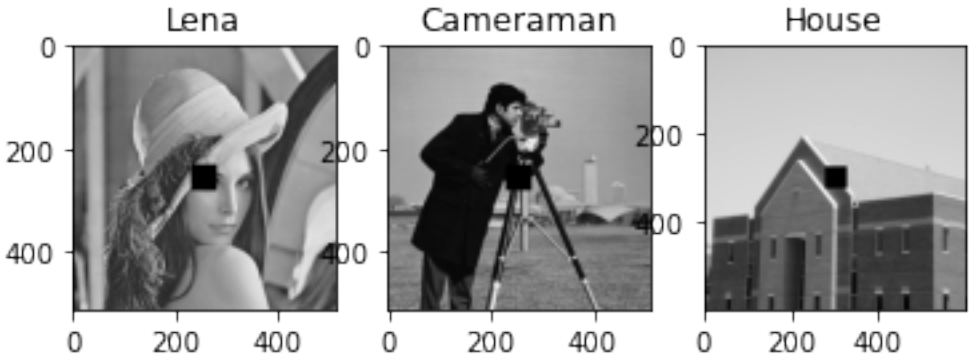


Exercicio D: Incluir 1 quadrado preto 15X15 no centro das imagens

Imagens originais



Imagens com quadrado preto no centro



<Figure size 432x288 with 0 Axes>