Fernando Soares (fjsoares@fc.ul.pt)

AULA PRÁTICA 3. *PRACTICAL CLASS 3*

1. OPERAÇÕES GEOMÉTRICAS. *GEOMETRIC OPERATIONS.*

1.1 Determine as dimensões de uma imagem, de 500x600, depois de uma rotação de 30^{O}. *Determine the dimensions of an image, 500x600, after a rotation of 30^{O}.*

1.2 Execute e analise o seguinte exemplo numérico de reamostragem. *Run and analyze the following resample numeric example.*

```
#%% Exercício numérico de reamostragem
v4 = np.array([[92, 44 ],
               [84, 152]])
x = np.array([[0, 1],
              [0, 1]])
y = np.array([[0, 0],
              [1, 1]])
dy = 0.6
dx = 0.1

#%% interpolação vizinho mais próximo
dist = np.sqrt((dx-x)**2+(dy-y)**2)
ind = np.argwhere(dist == np.min(dist))
z_viz = v4[ind[0, 0], ind[0, 1]]
print ('Reamostragem por vizinho mais próximo: ', z_viz)

#%% interpolação bilinear
k1 = dx*v4[0, 1]+(1-dx)*v4[0, 0]
k2 = dx*v4[1, 1]+(1-dx)*v4[1, 0]
z_bil = dy*k2+(1-dy)*k1
print ('Reamostragem por interpolação bilinear: ', int(z_bil))
# visualização
px = np.hstack((x[0, :], x[1, :]))
py = np.hstack((y[0, :], y[1, :]))
plt.figure()
plt.gca().invert_yaxis()
plt.plot(px, py, 'o'); plt.plot(dx, dy, 'or')
plt.text(dx, dy, 'PT')

#%% interpolacao bicúbica
xx = np.arange(4)
v16 = np.array([[79, 46, 120,  68],
                [88, 92,  44, 134],
                [91, 84, 152,  77],
                [31, 45, 130, 178]])

a = np.arange(4)*0.
plt.figure(figsize=(10, 2))
plt.subplot(121)
plt.title('Polinómios em x')
from scipy.interpolate import interp1d
for i in range(4):
    f = interp1d(xx, v16[i, :], kind='cubic')
    fcubic1 = f(np.arange(0, 3, 0.01))
    a[i] = fcubic1[int((1+dx)/0.01)]
    plt.plot(np.arange(0, 3, 0.01), fcubic1, 'g')
    plt.plot(xx, v16[i, :], 'o')

f = interp1d(xx, a, kind='cubic')
fcubic2 = f(np.arange(0, 3, 0.01))
z_bic = fcubic2[int((1+dy)/0.01)]
plt.subplot(122)
plt.plot(np.arange(0, 3, 0.01), fcubic2, 'g')
plt.plot(xx, a, 'o')
plt.title('Polinómio em y')
print ('Reamostragem por interpolação bicúbica: ', int(z_bic))
```

1.3 Programar a rotação da imagem **einstein01.tif**, com um dado ângulo, com a ajuda das linhas de código seguintes. Tire conclusões acerca do resultado obtido. *Program the rotation of the **einstein01.tif** image, with a given angle, with the help of the following lines of code. Draw conclusions about the result obtained.*

```
#%% Processo directo
# coordenadas iniciais dos pixels
xx = np.linspace(0, col1-1, col1)
yy = np.linspace(0, lin1-1, lin1)
x1, y1 = np.meshgrid(xx, yy)
# ângulo
ang = 20
alfarad = np.radians(ang)
# dimensoes da nova janela
col2 = int(math.ceil(abs(col1*np.cos(alfarad))+abs(lin1*np.sin(alfarad))))
lin2 = int(math.ceil(abs(lin1*np.cos(alfarad))+abs(col1*np.sin(alfarad))))
## novas coordenadas dos pixels após a rotação
x2 = x1*np.cos(alfarad)+y1*np.sin(alfarad)
y2 = -x1*np.sin(alfarad)+y1*np.cos(alfarad)
# translaccao para o primeiro quadrante
x2t = x2-np.min(x2)
y2t = y2-np.min(y2)
# sem translaccao para o primeiro quadrante
#x2t = x2
#y2t = y2
# imagem rodada
Rd = np.zeros((int(lin2), int(col2), dim[2]))
for k1 in range(0, dim[2]):
    for k2 in range(0, lin1):
        for k3 in range(0, col1):
            Rd[int(np.round(y2t[k2, k3])), int(np.round(x2t[k2, k3])), k1] = Img[k2, k3, k1]

plt.figure(figsize=(10, 5));
plt.imshow(np.uint8(Rd))
plt.title('Rotação: processo directo. Ângulo = '+str(alfarad*180/np.pi)+' graus')
plt.axis('off')
```

1.4 Executar a rotação usando a função *imrotate* com os vários métodos de reamostragem, e comparar resultados. *Run the rotation using the function imrotate with the various methods of resampling, and compare results.*

```
#%% Rotação com instrução do python
from skimage.transform import rotate
Rot = rotate(Img, ang, resize=True, order=1)
fig, ax = plt.subplots(1, 2)
ax[0].imshow(Img); plt.axis('off')
ax[1].imshow(Rot); plt.axis('off')
```

1.5 Executar a ampliação usando a função *imresize* com os vários métodos de reamostragem, e comparar resultados. *Run the magnification using the function imresize with the various methods of resampling, and compare results.*

```
#%% Ampliação com instrução do python
from skimage.transform import resize
Amp = resize(Img, [sy*lin1, sx*col1], mode='reflect', order=1)
plt.subplot(133); plt.imshow(Amp)
plt.title('Ampliação: com imresize')
```