Introduction to Computer Vision HW02

- Generating Hybrid Image -

2021-1학기 컴퓨터비전개론 059분반 201724437 김재현

Part 1.1

def boxfilter(n):

    assert n%2 == 1, "Dimesion must be odd"

    return np.ones((n,n))/(n\*\*2)

boxfilter(3)

**>> array([[0.11111111, 0.11111111, 0.11111111],**

**[0.11111111, 0.11111111, 0.11111111],**

**[0.11111111, 0.11111111, 0.11111111]])**

boxfilter(4)

**>> --------------------------------------------------------------------------- AssertionError**

**Traceback (most recent call last)**

**<ipython-input-43-5870f78beb34> in <module> ----> 1 boxfilter(4)**

**<ipython-input-42-5fc975c6d216> in boxfilter(n)**

**3 # If n is odd, create n \* n size array filled with 1. And divide it with sum of array, in this case , n\*\*2.**

**4 def boxfilter(n):**

**----> 5 assert n%2 == 1, "Dimesion must be odd" 6 return np.ones((n,n))/(n\*\*2)**

**AssertionError: Dimesion must be odd**

boxfilter(7)

**>> array([[0.02040816, 0.02040816, 0.02040816, 0.02040816, 0.02040816,**

**0.02040816, 0.02040816],**

**[0.02040816, 0.02040816, 0.02040816, 0.02040816, 0.02040816,**

**0.02040816, 0.02040816],**

**[0.02040816, 0.02040816, 0.02040816, 0.02040816, 0.02040816,**

**0.02040816, 0.02040816],**

**[0.02040816, 0.02040816, 0.02040816, 0.02040816, 0.02040816,**

**0.02040816, 0.02040816],**

**[0.02040816, 0.02040816, 0.02040816, 0.02040816, 0.02040816,**

**0.02040816, 0.02040816],**

**[0.02040816, 0.02040816, 0.02040816, 0.02040816, 0.02040816,**

**0.02040816, 0.02040816],**

**[0.02040816, 0.02040816, 0.02040816, 0.02040816, 0.02040816,**

**0.02040816, 0.02040816]])**

Part 1.2

def gauss1d(sigma):

    d = math.ceil(sigma\*6)//2\*2+1

    g1 = np.exp(-(np.arange(d)-d//2)\*\*2/(2\*sigma\*\*2))

    return g1/np.sum(g1)

gauss1d(0.3)

**>> array([0.00383626, 0.99232748, 0.00383626])**

gauss1d(0.5)

**>> array([0.10650698, 0.78698604, 0.10650698])**

gauss1d(1)

**>> array([0.00443305, 0.05400558, 0.24203623, 0.39905028, 0.24203623,**

**0.05400558, 0.00443305])**

gauss1d(2)

**>> array([0.0022182 , 0.00877313, 0.02702316, 0.06482519, 0.12110939,**

**0.17621312, 0.19967563, 0.17621312, 0.12110939, 0.06482519,**

**0.02702316, 0.00877313, 0.0022182 ])**

Part 1.3

def gauss2d(sigma):

    return np.outer(gauss1d(sigma), gauss1d(sigma))

gauss2d(0.5)

>> **array([[0.01134374, 0.08381951, 0.01134374],**

**[0.08381951, 0.61934703, 0.08381951],**

**[0.01134374, 0.08381951, 0.01134374]])**

gauss2d(1)

>> **array([[1.96519161e-05, 2.39409349e-04, 1.07295826e-03, 1.76900911e-03,**

**1.07295826e-03, 2.39409349e-04, 1.96519161e-05],**

**[2.39409349e-04, 2.91660295e-03, 1.30713076e-02, 2.15509428e-02,**

**1.30713076e-02, 2.91660295e-03, 2.39409349e-04],**

**[1.07295826e-03, 1.30713076e-02, 5.85815363e-02, 9.65846250e-02,**

**5.85815363e-02, 1.30713076e-02, 1.07295826e-03],**

**[1.76900911e-03, 2.15509428e-02, 9.65846250e-02, 1.59241126e-01,**

**9.65846250e-02, 2.15509428e-02, 1.76900911e-03],**

**[1.07295826e-03, 1.30713076e-02, 5.85815363e-02, 9.65846250e-02,**

**5.85815363e-02, 1.30713076e-02, 1.07295826e-03],**

**[2.39409349e-04, 2.91660295e-03, 1.30713076e-02, 2.15509428e-02,**

**1.30713076e-02, 2.91660295e-03, 2.39409349e-04],**

**[1.96519161e-05, 2.39409349e-04, 1.07295826e-03, 1.76900911e-03,**

**1.07295826e-03, 2.39409349e-04, 1.96519161e-05]])**

Part 1.4.a : python 소스코드 참조(line 124)

Part 1.4.b: python 소스코드 참조(line 152)

Part 1.4.c, d

dogIm = Image.open('2b\_dog.bmp')

dogIm = dogIm.convert('L')

dogImArray = np.asarray(dogIm)

dogImGaussArray = gaussconvolve2d(dogImArray, 3)

dogImGaussArray = dogImGaussArray.astype('uint8')

dogImGauss = Image.fromarray(dogImGaussArray)

dogImGauss.save('dog\_gaussian.bmp', 'PNG')

dogIm.show()

dogImGauss.show()

>>

그림 1,2 –2b\_dog.bmp, dog\_gaussian.bmp

Part 2.1 (함수 imageFromChArray, colorLowPassChannels는 코드 참조; line 191~222)

dogIm = Image.open('2b\_dog.bmp')

dogLowPassIm = imageFromChArray(colorLowPassChannels(dogIm, 3))

dogLowPassIm.show()

dogLowPassIm.save("dog\_lowpass.bmp", "PNG")

>>



그림 3 – dog\_lowpass.bmp

Part 2.2 (함수 colorHighPassChannels는 코드 참조; line 250~268)

catIm = Image.open('2a\_cat.bmp')

catHighPassIm = imageFromChArray(colorHighPassChannels(catIm, 3, 1))

catHighPassIm.show()

catHighPassIm.save("cat\_highpass.bmp", "PNG")

>>

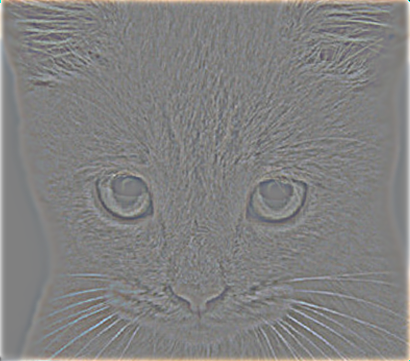


그림 4 – cat\_highpass.bmp

Part 2.3 (함수 colorHybrid는 코드 참조; line 294~314)

catDogHybridIm  = imageFromChArray(colorHybrid(dogIm, catIm, 3, 3))

catDogHybridIm.show()

catDogHybridIm.save("cat\_dog\_hybrid.bmp","PNG")

>>



그림 5 – cat\_dog\_hybrid.bmp